Stefan Katzenbeisser, Volkmar Lotz, Edgar Weippl (Hrsg.)

Sicherheit 2014

Sicherheit, Schutz und Zuverlässigkeit

19.-21. März 2014
in Wien

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

Volume P-228

ISBN 978-3-88579-622-0
ISSN 1617-5468

Volume Editors
Stefan Katzenbeisser
Technische Universität Darmstadt
Hochschulstraße 10, D-64289 Darmstadt
E-Mail: katzenbeisser@seceng.informatik.tu-darmstadt.de
Volkmar Lotz
SAP Labs France
805 Avenue du Dr. Maurice Donat, BP1216, F-06254 Mougins Cedex
E-Mail: volkmar.lotz@sap.com
Edgar Weippl
SBA Research
Favoritenstraße 16, A-1040 Wien
E-Mail: eweippl@sba-research.org

printed by Köllen Druck+Verlag GmbH, Bonn

Vorwort

Die Fachtagung *Sicherheit, Schutz und Zuverlässigkeit* des Fachbereichs *Sicherheit – Schutz und Zuverlässigkeit* der Gesellschaft für Informatik fand in der siebenten Ausgabe in Wien statt. Die SICHERHEIT 2014 bot einem breiten Publikum aus Forschung, Entwicklung und Anwendung einen Einblick in aktuelle Forschungsfragen der Sicherheit und Zuverlässigkeit von IT-basierten Systemen.

Der vorliegende Tagungsband umfasst 35 Beiträge, die aus insgesamt 85 Einreichungen ausgewählt wurden. Traditionsgemäß durften Beiträge sowohl in Deutsch als auch in Englisch eingereicht werden. Dieses Jahr wurde erstmals ein eigener *practitioners track* eingerichtet, für den 11 Einreichungen aus der Wirtschaft ausgewählt wurden, die Resultate aus industrieller Forschung und Entwicklung besprechen, durchgeführte Studien dokumentieren oder Berichte über "best practices" enthalten. Neben den regulären Beiträgen konnten wir Dr. Michael Schmitt (SAP), Prof. Michael Huth (Imperial College London) und Prof. Reinhard Posch (TU Graz) für eingeladene Vorträge gewinnen.

Dieser Tagungsband enthält zudem 3 Arbeiten des Western European Workshop on Research in Cryptology (WEWoRC 2013), der von der Fachgruppe Angewandte Kryptographie der GI im Juli 2013 in Karlsruhe organisiert wurde.

Unser Dank gilt allen, die sich am Gelingen der SICHERHEIT 2014 beteiligt haben. Allen voran sind natürlich hier die Autorinnen und Autoren, die Mitglieder des Programmkomitees sowie die externen Gutachter zu nennen. Unser besonderer Dank gilt Yvonne Poul und Niklas Büscher, die mit der lokalen Organisation der Konferenz bzw. der Erstellung dieses Proceedingsbandes betraut waren. Nicht zuletzt gilt unser Dank auch dem Leitungsgremium des GI-Fachbereichs Sicherheit, Schutz und Zuverlässigkeit, das uns tatkräftig in der Organisation der Konferenz unterstützt hat.

Stefan Katzenbeisser
Volkmar Lotz
Edgar Weippl

Februar 2014

Tagungsleitung

Hannes Federrath (Sprecher des Fachbereichs, Universität Hamburg)
Stefan Katzenbeisser (Program Co-Chair, Technische Universität Darmstadt)
Volkmar Lotz, (Program Co-Chair, SAP Labs France)
Michael Waidner (Technische Universität Darmstadt & Frauenhofer SIT)
Edgar Weippl (General Chair, SBA Research)

Lokale Organisation

Niklas Büscher (Technische Universität Darmstadt, Germany)
Yvonne Poul (SBA Resarch, Austria)

Programmkomitee

Ammar Alkassar (Sirrix AG, Germany)
Frederik Armknecht (Universität Mannheim, Germany)
Bernhard Beckert (KIT Karlsruhe, Germany)
Ruth Breu (Universität Innsbruck, Austria)
Rainer Boehme (University of Münster, Germany)
Jens Braband (Siemens AG, Germany)
Arslan Broemme (Vattenfall GmbH, Germany)
Christoph Busch (Hochschule Darmstadt, Germany)
Christian Cachin (IBM Research - Zurich, Switzerland)
Peter Dencker (ETAS Stuttgart, Germany)
Jana Dittmann (Uni Magdeburg, Germany)
Claudia Eckert (TU München & Fraunhofer AISEC, Germany)
Dominik Engel (Salzburg University of Applied Sciences, Austria)
Bernhard Fechner (Universität Augsburg, Germany)
Hannes Federrath (Universität Hamburg, Germany)
Ulrich Flegel (Hochschule für Technik Stuttgart, Germany)
Felix C. Freiling (Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany)
Walter Fumy (Bundesdruckerei GmbH, Germany)
Willi Geiselmann (KIT Karlsruhe, Germany)
Rüdiger Grimm (Universität Koblenz-Landau, Germany)
Karl-Erwin Grosspietsch
Marit Hansen (Unabhängiges Landeszentrum für Datenschutz, Schleswig-Holstein, Germany)
Dieter Hutter (DFKI/Universität Bremen, Germany)
Jan Jürjens (TU Dortmund, Germany)

Stefan Katzenbeisser (Technische Universität Darmstadt, Germany)
Hubert Keller (KIT Karlsruhe, Germany)
Herbert Klenk (Cassidian/EADS Manching, Germany)
Klaus-Michael Koch (Technikon, Austria)
Ioannis Krontiris (Universität Frankfurt, Germany)
Ralf Kuesters (University of Trier, Germany)
Volkmar Lotz (SAP Labs, France)
Stefan Lucks (Bauhaus-University Weimar, Germany)
Michael Meier (Universität Bonn & Fraunhofer FKIE, Germany)
Alexander Nouak (Fraunhofer IGD, Germany)
Michael Nüsken (b-it cosec Bonn, Germany)
Isabel Münch (Bundesamt für Sicherheit in der Informationstechnik, Germany)
Frank Ortmeier (Universität Magdeburg, Germany)
Sebastian Pape (TU Dortmund, Germany)
Günther Pernul (Universität Regensburg, Germany)
Kai Rannenberg (Universität Frankfurt, Germany)
Peer Reymann (ITQS Gesellschaft für Qualitätssicherung in der Informationstechnologie mbH)
Konrad Rieck (Universität Göttingen, Germany)
Stefanie Rinderle-Ma (Universität Wien, Austria)
Heiko Rossnagel (Fraunhofer IAO, Germany)
Francesca Saglietti (Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany)
Andreas Schaad (SAP AG, Germany)
Ingrid Schaumäller (FH OÖ, Campus Hagenberg, Austria)
Sebastian Schmerl (AGT International, Switzerland)
Matthias Schunter (Intel Labs, Germany)
Jörg Schwenk (Ruhr-Universität Bochum, Germany)
Jean-Pierre Seifert (TU Berlin, Germany)
Claus Stark (Citigroup Global Markets Deutschland AG, Germany)
Mark Strembeck (WU Wien, Austria)
Thorsten Strufe (Technische Universität Darmstadt, Germany)
Claus Vielhauer (Otto-von-Guericke University Magdeburg & Brandenburg University of Applied Sciences, Germany)
Melanie Volkamer (Technische Universität Darmstadt & CASED, Germany)
Hans von Sommerfeld (VOSDAV GmbH Berlin und Sprecher GI FG EZQN, Germany)
Edgar Weippl (Vienna University of Technology, Austria)
Stefan Weiss (Swiss Re, Switzerland)
Steffen Wendzel (Fraunhofer FKIE, Bonn, Germany)
Bernhard C. Witt (it.sec GmbH & Co. KG, Germany)
Christopher Wolf (Ruhr-Universität Bochum, Germany)
Xuebing Zhou (Hochschule Darmstadt, Germany)

Zusätzliche Gutachter

Farzaneh Abed
Daniel Arp
Florian Böhl
Gkhan Bal
Steffen Bartsch
Thomas Bauereiß
Nico Döttling
Adrian Dabrowski
Andreas Dewald
Michael Diener
Manfred Erjak
Daniel Fett
Daniel Fitzner
Christian Forler
Karl-Peter Fuchs
Ludwig Fuchs
Matthias Gabel
Matthias Gander
Hugo Gascon
Markus Gattol
Christoph Gerber
Simon Greiner
Florian Hahn
Mohammad Halawah
Marvin Hegen
Ronald Heinrich
Dominik Herrmann
Tobias Hildebrandt
Milen Hringov
Markus Huber
Angela Jaeschke
Basel Katt
Peter Kieseberg
Vladimir Klebanov
Sascha Koschinat
Simone Kriglstein
Katharina Krombholz
Michael Kubach
Sebastian Kurowski
Martin Lambertz
Stefan Laube
Maria Leitner
Eik List
Daniel Loebenberger
Malte Möser
Stefan Meier
Vasily Mikhalev
Martin Mulazzani
Michael Netter
Christian Neureiter
Giang Nguyen
Sebastian Nielebock
Tobias Nilges
Eray Özmü
Andreas Reisser
Christian Reuter
Markus Riek
Stefanie Roos
Johannes Sänger
Ahmad Sabouri
Enrico Scapin
Sigrid Schefer-Wenzl
Sebastian Schinzel
Guido Schmitz
Pascal Schoettle
Bahareh Shojaie
Christian Sillaber
Barbara Sprick
Benjamin Stritter
Arnold Sykosch
Andreas Tomandl
Thomas Trojer
Markus Tschersich
Johanna Ullrich
Armin Veichtlbauer
Nicolas Von Zur Mühlen
Tobias Wahl
Jakob Wenzel
Lars Wolos
Christian Wressnegger
Fabian Yamaguchi
Philipp Zech
Viviane Zwanger

Inhaltsverzeichnis

Angewandte Kryptographie

Multimedia Sicherheit

E-Commerce und Mobile Anwendungen

Forensik

Zuverlässige Systeme & Kritische Infrastrukturen

Western European Workshop on Research in Cryptology 2013

Continuous Authentication on Mobile Devices by Analysis of Typing Motion Behavior

Hugo Gascon[1], Sebastian Uellenbeck[2], Christopher Wolf[2], Konrad Rieck[1]

[1]Computer Security Group
University of Göttingen
Goldschmidtstrasse 7
37073 Göttingen
{hgascon, konrad.rieck}@uni-goettingen.de

[2]Horst Görtz Institute for IT-Security
Ruhr-University Bochum
Universitätsstrasse 150
44810 Bochum
{sebastian.uellenbeck, christopher.wolf}@rub.de

Abstract: Smartphones have become the standard personal device to store private or sensitive information. Widely used as every day gadget, however, they are susceptible to get lost or stolen. To protect information on a smartphone from being physically accessed by attackers, a lot of authentication methods have been proposed in recent years. Each one of them suffers from certain drawbacks, either they are easy to circumvent, vulnerable against shoulder surfing attacks, or cumbersome to use. In this paper, we present an alternative approach for user authentication that is based on the smartphone's sensors. By making use of the user's biometrical behavior while entering text into the smartphone, we transparently authenticate the user in an ongoing-fashion. In a field study, we asked more than 300 participants to enter some short sentences into a smartphone while all available sensor events were recorded to determine a typing motion fingerprint of the user. After the proper feature extraction, a machine learning classifier based on Support Vector Machines (SVM) is used to identify the authorized user. The results of our study are twofold: While our approach is able to continuously authenticate some users with high precision, there also exist participants for which no accurate motion fingerprint can be learned. We analyze these difference in detail and provide guidelines for similar problems.

1 Introduction

Smartphones have become the epitome of the *always on, always connected* trend. They combine and extend the functionality of feature phones with a multitude of characteristics from desktop computers, and this have made them a world-wide commodity with an ever increasing market share. Unfortunately, the great amount of personal data stored on these devices, including passwords or banking information, have made them an attractive target

for criminals. To prevent this information from being physically retrieved by attackers, access control mechanisms like user authentication have been proposed. When focusing on the Android OS, user authentication can mainly be divided into two categories.

First, methods that are based on knowledge like PIN, password, and Android Unlock Patterns. Second, biometrics based methods. It is well known that knowledge-based authentication factors, such as passwords and PINs, provide only limited security on smartphones. Similarly, some biometric factors, such as Face Unlock, also fail to provide a reliable user authentication (c. f., [MDAB10, BPA12, UDWH13, FM12]). In either case: A smartphone that was unlocked once stays unlocked until it is actively locked again or a fixed period of time elapses without interaction. Therefore, there always exists a time frame in which an attacker can steal the unlocked phone to later obtain all data stored on the device.

Our Contribution In this paper, we present a user authentication approach that is based on analyzing the typing motion behavior of a specific user. To counter attacks that happen after the smartphone is unlocked, our approach uses a continuous authentication scheme by means of biometry while the user is entering text. In general, biometric features can be divided into the two classes: physiological and behavioral biometrics. Physiological biometrics rely on something *the user is*. This class contains features like fingerprint [CKL03], face [ZCPR03], hand geometry, voice, and iris. They are often used for authentication to high secure entrance systems [JRP06] but need special hardware to be identifiable. Apart from that, behavioral biometrics are based on how *the user behaves*, like keystroke patterns, the user's gait [GS09], or location information without requiring additional hardware. This class can be used to authenticate the user uninterruptedly and is also implementable with built-in smartphone sensors.

To gather behavioral biometrics, we have developed a software keyboard application for the Android OS that stores all sensor data for further learning and evaluation. In a field study, we asked more than 300 participants to enter a short text into our smartphone. To ensure that only motion data related to typing behavior is used to learn a profile, we preprocessed the sensor signals according to certain time constraints. Next, we extracted various features leading to a 2376-dimensional vector representing the typing motion behavior of a user in a given time frame. By means of SVMs, we learn a classifier in order to identify the behavioral fingerprint of each user. In the end, we analyze the results of our approach and its limitations, and discuss what obstacles can be found when tackling similar problems and their possible improvements.

Summary In summary, we make the following contributions:

- We implemented a software keyboard for Android that enabled us to collect the typing behavior of 300 users in a field study.
- We designed a time-based feature extraction method and evaluated the performance of a machine learning classifier in a continuous authentication problem.
- Finally, we discuss limitations to sensor-based approaches in authentication problems and propose several improvements and considerations.

2 Related Work

Besides classic user authentication on smartphones by means of PINs or passwords, there has been a lot of work in this area in recent years. Frank et al. [FBM+13] evaluated the feasibility of continuous user authentication by means of touchscreen features. Although they confirmed that user authentication is possible despite having a False Negative Rate of only 0% to 4%, they do not consider other features obtained by motion or position sensors.

Zhen et al. [ZBHW12] proposed an extension to the plain PIN authentication method strengthened with sensor data and timings. They collected five different acceleration values, the touching pressure, the touched area on the screen, and different time values like key-hold time or inter-key time. They evaluated previously specified 4-digit and 8-digit PINs from over 80 participants. As a result they achieve an equal error rate between 3.65% and 7.34% referred to the predefined PIN. In contrast to them, we propose a continuous authentication methodology intended to be used without taking predefined text into consideration. After a learning phase, users are authenticated while entering normal text.

Sandnes and Zhang [SZ12] studied strategies for identifying users based on touch dynamics. They monitor and extract features like left vs. right hand dominance, one-handed vs. bimanual operation, stroke size, stroke timing, symmetry, stroke speed, and timing regularity. In an experiment with 20 participants, they found their approach and feature set useful. In contrast to us, they did not consider the behavior of attackers and had a very limited set of participants. They could therefore not give any numbers on how this approach performs in reality. Shi et al. [SNJC10] investigated implicit authentication through learning the behavior of a user. They created user patterns based on sent and received text messages and phone calls, the browser history, and the location of the smartphone. This approach also allowed for considering typing behavior whereas this was only a theoretic but not reviewed aspect of their work.

De Luca et al. [DHB+12] used biometric features to improve the security of the Android Unlock screen. While performing the wipe gesture to unlock the smartphone's screen, they collected all data available from the touchscreen, including pressure, size, coordinates, and time. In a study with 48 participants, they could show that their approach reaches a 98% True Positive Rate but comes also with the price of a 43% False Positive Rate for the best case.

Li et al. [LZX13] proposed a system similar to our approach, that uses a continuous authentication for smartphones by taking the user's finger movement pattern into account for learning. In contrast to us, they did not consider entering text into a softkeyboard but only gestures like sliding towards a special direction or taps.

Keystroke dynamics have also been considered for desktop computers [MR00, PKW04, UW85, ASL+04]. Since smartphones comprise sensors to capture environmental changes, they offer more capabilities to authenticate users. In the following, we show that sensor data can be used to gather typing motion behavior.

3 Methodology

In this section we describe the different steps of our method. First, we present how the user data is collected by means of our softkeyboard prototype and then the feature building process is described. These features allow us to design a learning and classification setup that lets us identify a user during the ongoing interaction with the device.

3.1 Data Collection

Smartphones comprise many sensors that can be used to evaluate real-world characteristics, like user biometrics. Still they have to been processed through adequate software beforehand. The majority of smartphones do not contain hardware keyboards to enter characters but a touchscreen that can be utilized as keyboard. Here, software displays a virtual keyboard (softkeyboard) on the touchscreen and the user's input is forwarded by the touchscreen to the underling application.

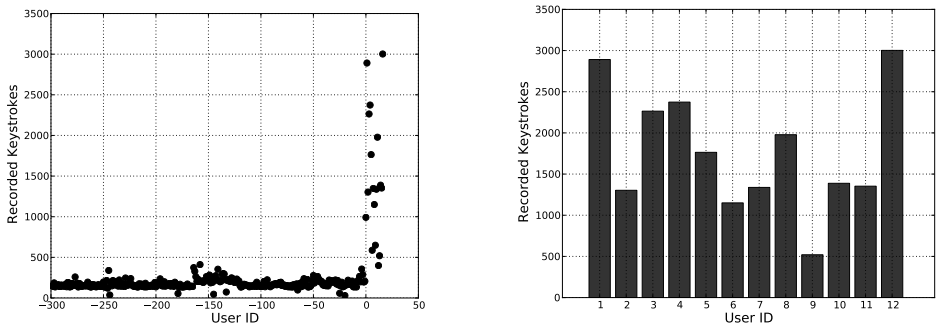
In this paper, our goal is to explore the possibility of learning an individual profile from the motion behavior of a smartphone while the user is typing. This model will then be used to continuously authenticate the user against unauthorized attackers. To this aim and in order to collect the user's motion data, we have developed a softkeyboard prototype for the Android OS being able to read and store all sensor events for further analysis.

The softkeyboard makes use of the accelerometer to measure acceleration, the gyroscope to measure torque and the orientation sensor to measure the relative position of the device. Each one of these parameters is measured in the three dimensions of space x, y, and z. Additionally, our prototype utilizes the display to get the exact position the user has touched to introduce a keystroke. On the software-side, a keystroke is divided into three events: *onPress*, *onKey*, and *onRelease*. The first is raised when the user touches the screen, the second when the software sends the character to the underlaying layer, and the last presents the event when the user's finger leaves the screen. This fine-grained approach is necessary since smartphones allow to change the selected character after the *onPress* event so the final character is only fixed when the user's finger releases the screen. For each of this three events, we record the timestamp with a granularity of milliseconds and all sensor values. Since the duration of a keystroke can—depending on the user—last from 80 ms to 500 ms, we record all sensor events that occur while a user is entering text.

In order to analyze the typing motion behavior, we asked 315 people to write a short and predefined text ($\approx$ 160 characters) on the test smartphone using our softkeyboard prototype. The text introduced by the subjects was a set of different *pangrams* containing all letters of the English language like *The quick brown fox jumps over the lazy dog*. By choosing such sentences we assured that the users had to touch each virtual character at least once. When asking them to support our work we explained that their data was taken and recorded for a scientific experiment.

We split the keystroke data into two fractions. The first fraction was taken from 303 participants that entered the text only once. This group represents individuals who in a figurative attack scenario, take the device away while unlocked. Their typing motion behavior is

therefore considered as an attempt to interact with the device unauthorized. The second fraction, the remaining 12 participants, were asked more than 10 times to write the same short text. This group was intended to be considered as authorized users. Their larger amount of data should allow us to model a unique profile that can be used to individually distinguish their behavior from the one of any other unauthorized attacker. Figure 1(a) shows the total number of keystrokes introduced by each user. In the following section we present the preprocessing steps to extract relevant features from the sensor signals recorded in real time and which will be fed to the classifier.



(a) Keystrokes per User

(b) Keystrokes per Authorized User

Figure 1: Total number of keystrokes introduced per unauthorized and authorized users.

3.2 Time-based Feature Extraction

During the data collection phase, different users introduce text through the softkeyboard while motion data is continuously recorded by the sensors. In order to ensure that only motion information related to typing behavior is used to learn each user profile, the sensor signals need to be processed according to certain time constraints.

The goal during the feature extraction phase is to compute a data point from all values acquired during T seconds while a user is typing. Figure 2 shows how different timers are used to correctly identify and record the data captured by the sensors.

Once a user has started typing, a data point is computed every T seconds. If the user does not introduce any keystroke during T_{stop} seconds, we assume that the user has stopped typing. This allows us to discard sensor values measured after T_1. In case that recorded values do not sum up to T seconds and the user resumes typing, new sensor values will be added to the previously captured up to complete the T seconds. Since data points are computed on a time basis, the burst of keystrokes used to compute each data point may be different and depends on the typing speed and pauses performed by the user.

Captured signals are then normalized to remove the artifacts or constant values such as the $9.81\ m/s^2$ typically measured in the accelerometer z coordinate as result of gravity. The normalization is performed in a similar manner as described by Aviv et al. [ASBS12].

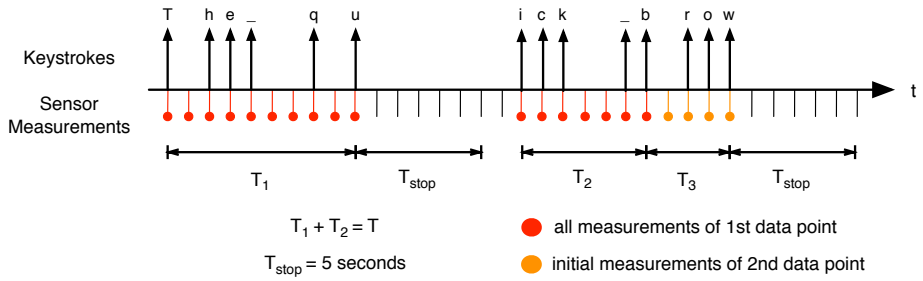


Figure 2: Description of the relation between typed keystrokes and motion measurements. Certain timing conventions are introduced to slice sensor signals according to a predefined time window.

As mentioned above, the 9 signals resulting from the spatial readings of the accelerometer, gyroscope, and orientation sensors are recorded during T seconds. Three normalized forms are computed for each one of them. Table 1 shows how a mean, linear, and 3-degree spline (P_{3d}) are subtracted from the original values to obtain the three normalized versions that will be used to extract a set of discrete features.

Table 1: Description of the 3 normalization forms applied to each sensor signal.

Type	**Fit**	**Normalization**
Mean	$m = \frac{1}{N}\sum_1^N s_i$	$S_m = S - m$
Linear	$mx + c$	$S_l = S - (mx + c)$
Spline	P_{3d}	$S_p = S - P_{3d}$

Figure 3 presents a visual example of one of the sensor signals. The three types of fit reconstructions are over imposed to the original signal. Their respective subtraction in each one of the cases will produce the normalized signals S_m, S_l and S_P. The set of features described in Table 2 are then extracted from each sequence of normalized values.

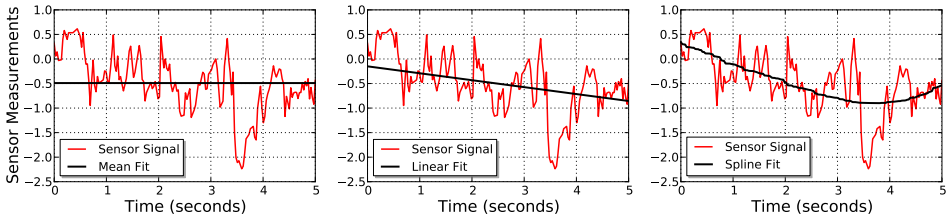


Figure 3: Example of accelerometer sensor readings for T = 5 in the x direction and the corresponding three types of fit reconstructions.

Table 2: Set of features extracted from each one of the normalized sensor signals.

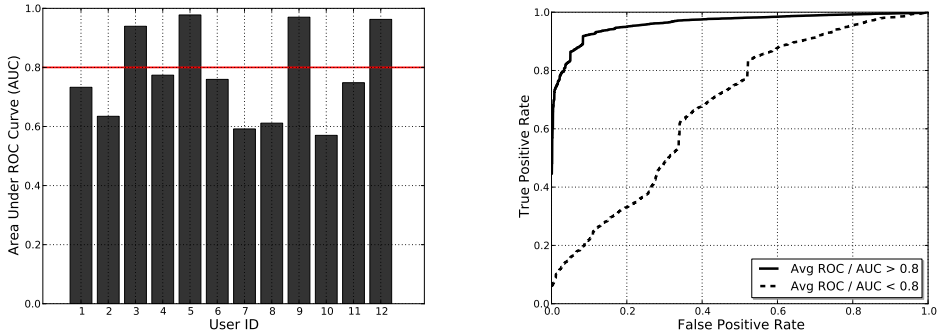
Features	Length	Description
Simple Statistics	6	root mean square, mean, standard deviation, variance, max and min
Spline Coefficients	6	coefficients of a 5-degree smoothing spline
Spline Simple Statistics	6	simple statistics from 5-degree smoothing spline
iFFT Spline Features	35	inverse DFT of a DFT of the 5-d spline fit curve using 35 samples
iFFT Signal Features	35	inverse DFT of a DFT of the sensor signal using 35 samples

The total number of features extracted from each normalized signal is therefore 88. As the signals from 9 sensors are recorded simultaneously and 3 normalize versions are reconstructed, the total number of features computed is $3 \times 9 \times 88 = 2376$. As a result, a 2376-dimensional vector, representing a unique data point, is computed every T seconds when the user is typing.

4 Evaluation

Our aim is to be able to correctly identify the profile of an authorized user versus that of several unauthorized attackers. Therefore, we pose a two-class classification problem. The first class, labeled $+1$, is formed by the data points generated from the signals of one of the 12 users with the largest amount of recorded data. The second class, labeled -1, is formed by the data points generated from the typing behavior of the 303 users that have introduced a short sequence of characters. A linear Support Vector Machine (SVM) is finally used to find the hyperplane that optimally splits both classes of data points in the 2376-dimensional space spanned by the generated featured vectors.

To evaluate the classification performance, we compute the Receiver Operating Characteristics (ROC) curve for the classifier of each user. The ROC curve presents the performance of the classifier in terms of *true positive rate* (TPR) versus *false positive rate* (FPR), where the area under the curve (AUC) of a perfect classifier has a value of 1. We can observe in Figure 4(a) that the classifiers learnt for the different users show a heterogeneous performance. While users with ID 3, 5, 9 and 12 can be identified with an AUC above 0.9, the AUC remains under 0.6 for some of the other users. These results suggest that only a certain number of users have a distinct and characteristic profile. We have established that a user requires a classification AUC above 0.8 to be identified with sufficient confidence. Now we can distinguish between identifiable and non-identifiable users. Figure 4(b) presents the average ROC for all of the users in both groups. The average classification performance of the classifiers for non-identifiable users presents a considerable high FPR of 35%, while reaching only a TPR of 58% at this point. On the contrary, if we consider the group of clearly identifiable users, the average classification performance achieves a TPR of 92%, reached at only 1% of FPR.



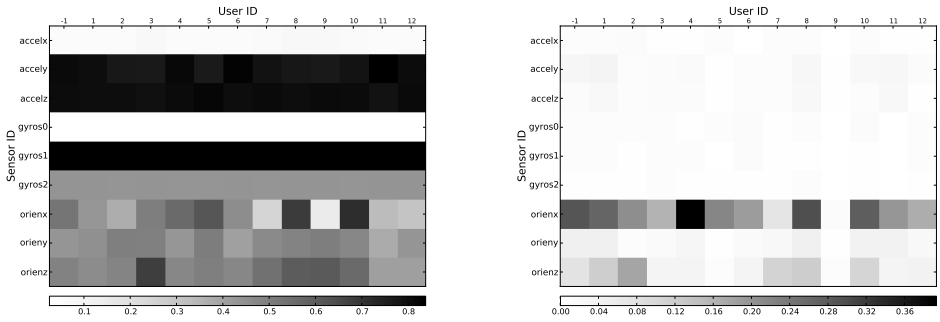
(a) AUC per user

(b) Averaged ROCs

Figure 4: Classification performance for each authorized user. Figure 4(b) shows both, the average ROC of users with an individual AUC over 0.8 and the average ROC of users with an individual AUC under 0.8.

5 Discussion

In order to explain such an uneven performance and also to see what makes a user identifiable, we observe the original sensor measurements for each user. Figure 5 presents the sample mean and standard deviation of the normalized values recorded by each sensor. It can be noticed that values recorded by the accelerometer and gyroscope sensors have very similar average values with a very small dispersion for all the users, including the class of unauthorized users labeled -1. In fact, we see that the largest differences between the typing motion behavior of the users is mainly characterized by the orientation sensor. In particular, the average from the values recorded in the x coordinate presents not only the higher differentiation between users but also a higher and disparate dispersion.



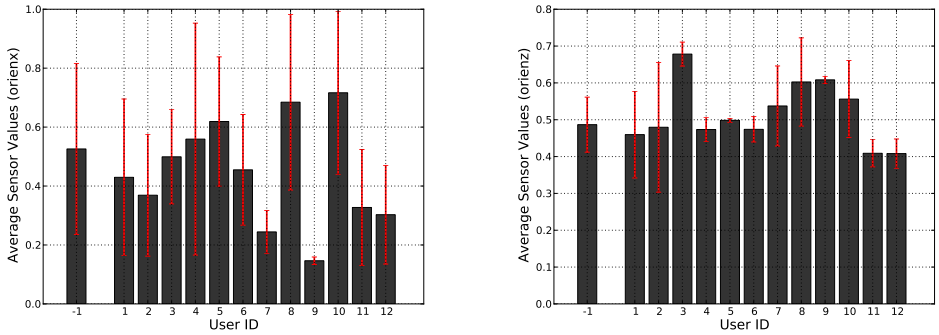
(a) Mean of sensor values

(b) Standard deviation of sensor values

Figure 5: Mean and standard deviation of the sensor values recorded from each user and normalized between 0 and 1. User ID -1 represents all measurements corresponding to users with negative IDs (attackers).

It should be noted that although the mean and standard deviation of the sensor values can provide certain information about the dataset and the user profiles, the SVM is able to determine more complex relationships within the data in the high-dimensional feature space built from the original sensor values. This may be difficult to explain but led to better results. Nevertheless, these simple statistics already allow us to draw some clear conclusions on the good classification performance of the identifiable users.

For example, Figure 6(a) shows how user 9 presents a very distinctive behavior with the lowest mean value and a extremely low standard deviation in the x coordinate of the orientation sensor. This is very interesting as this user have introduced a smaller amount of data. In the same way, Figure 6(b) shows how user 3 presents a very high mean value on the z coordinate of the orientation sensor, while having also a very low standard deviation. We thus conclude that the high variance on the x coordinate of the orientation is a root-cause for the differences in authentication performance. Likely, some users seem to randomly shake or shift the smartphone during typing on this particular axis and thereby limit the learning of an accurate behavioral fingerprint.



(a) Orientation sensor, x coordinate

(b) Orientation sensor, z coordinate

Figure 6: Mean and standard deviation of measurements recorded by the accelerometer in the x and z coordinates for all users.

In spite of the uneven performance, there exist some considerations that we have taken into account to assure that the obtained results are sound, and also some remarks that need to be addressed by any research dealing with similar problems to the one described in this paper. In the first place, in every experiment that measures human behavior and therefore requires human test subjects, the environmental conditions during the collection phase need to be extremely well controlled. As reported in previous research, even the behavior of a unique person can greatly differ from one day to another. Consequently, not only different measurements from the same person have been taken on different days, but in a well conditioned setting in order to minimize the influence of the environment in the captured data.

Additionally, a particular issue may arise when dealing with time series. In order to select the best parameters for the learnt model during the training phase a typical cross-validation strategy can be followed. However, the time dependence of the data imposes certain con-

straints during the phases of training and validation which are often overlooked. For instance, in a k-fold cross validation, the data points in each fold should not be randomized. This may improve the final performance but would also be an unrealistic setup if learning is done on a real device while the user is introducing text. If we consider each sequence of characters as independent from each other, cross-validation folds may be shuffled as blocks but the data points that form each one of them should never be randomized. Furthermore, the data points used for testing need to lie in the future of the training and validation sets for the same reason. Moreover, the non-trivial problem of selecting the best possible parameters to model each user can led to a decrease in performance. An extensive analysis of the influence of parameters like T_{stop}, the degree for spline reconstructions of the signals or the length of the different FFT could reveal better combinations of values to identify each individual user.

6 Conclusions and Future Work

In recent years, many different solutions to security problems but also attacks involving the use of smartphone sensors have been proposed. The standard adoption of touchscreens in these devices has led to a new strain of biometric related research, aiming in most cases at improving the traditional authentication scheme of text-based passwords. As the continuous interaction of users with these devices allows for a continuous analysis of their behavior, we have explored in this paper how the motion information retrieved by the sensors can be used to build a unique typing motion profile of an authorized user. During the evaluation and discussion sections several conclusions have been drawn regarding the achieved performance. We have shown that certain users show a characteristic behavior which can be identified with a TPR of 92% at a FPR of 1%, while some other users are hardly distinguishable from each other and from the typing motion behavior of the attackers.

These results and the proposed method open the door to future improvements. For instance, we have posed the problem as a binary classification experiment, however other strategies may come to mind. In the general case or if attacker data is scarce, an anomaly detection setup can be implemented, where the authorized user is modeled using a one-class SVM. Moreover, in this type of problem, the false positive rate should be minimized to avoid constant lockings of the device and additional requests to re-authorize the user with text or graphical passwords. Certain techniques as majority voting or making the classifier decide that a data point belongs to an unauthorized user only after a number of data points have been identified as an attack, are interesting paths which we plan to explore in our future work.

References

[ASBS12] Adam J. Aviv, Benjamin Sapp, Matt Blaze, and Jonathan M. Smith. Practicality of Accelerometer Side Channels on Smartphones. In Robert H'obbes' Zakon, editor, *ACSAC*, pages 41–50. ACM, 2012.

[ASL+04] Lívia C. F. Araújo, Luiz H. R. Sucupira, Miguel Gustavo Lizárraga, Lee Luan Ling, and João Baptista T. Yabu-uti. User Authentication through Typing Biometrics Features. In David Zhang and Anil K. Jain, editors, *ICBA*, volume 3072 of *Lecture Notes in Computer Science*, pages 694–700. Springer, 2004.

[BPA12] Joseph Bonneau, Sören Preibusch, and Ross Anderson. A Birthday Present Every Eleven Wallets? The Security of Customer-Chosen Banking PINs. In Angelos D. Keromytis, editor, *Financial Cryptography*, volume 7397 of *Lecture Notes in Computer Science*, pages 25–40. Springer, 2012.

[CKL03] T. Charles Clancy, Negar Kiyavash, and Dennis J. Lin. Secure Smartcard-based Fingerprint Authentication. In *ACM SIGMM Workshop on Biometric Methods and Applications*, pages 45–52, 2003.

[DHB+12] Alexander De Luca, Alina Hang, Frederik Brudy, Christian Lindner, and Heinrich Hussmann. Touch Me Once and I Know it's You! Implicit Authentication Based on Touch Screen Patterns. In Joseph A. Konstan, Ed H. Chi, and Kristina Höök, editors, *CHI*, pages 987–996. ACM, 2012.

[FBM+13] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. Touchalytics: On the Applicability of Touchscreen Input as a Behavioral Biometric for Continuous Authentication. *IEEE Transactions on Information Forensics and Security*, 8(1):136–148, 2013.

[FM12] Rainhard Dieter Findling and Rene Mayrhofer. Towards Face Unlock: On the Difficulty of Reliably Detecting Faces on Mobile Phones. In Eric Pardede and David Taniar, editors, *MoMM*, pages 275–280. ACM, 2012.

[GS09] Davrondzhon Gafurov and Einar Snekkenes. Gait Recognition Using Wearable Motion Recording Sensors. *EURASIP J. Adv. Sig. Proc.*, 2009, 2009.

[JRP06] Anil K. Jain, Arun Ross, and Sharath Pankanti. Biometrics: A Tool for Information Security. *IEEE Transactions on Information Forensics and Security*, 1(2):125–143, 2006.

[LZX13] Lingjun Li, Xinxin Zhao, and Guoliang Xue. Unobservable Re-authentication for Smartphones. In *NDSS*. The Internet Society, 2013.

[MDAB10] Steven J. Murdoch, Saar Drimer, Ross J. Anderson, and Mike Bond. Chip and PIN is Broken. In *IEEE Symposium on Security and Privacy*, pages 433–446. IEEE Computer Society, 2010.

[MR00] Fabian Monrose and Aviel D. Rubin. Keystroke Dynamics as a Biometric for Authentication. *Future Generation Comp. Syst.*, 16(4):351–359, 2000.

[PKW04] Alen Peacock, Xian Ke, and Matthew Wilkerson. Typing Patterns: A Key to User Identification. *IEEE Security & Privacy*, 2(5):40–47, 2004.

[SNJC10] Elaine Shi, Yuan Niu, Markus Jakobsson, and Richard Chow. Implicit Authentication through Learning User Behavior. In Mike Burmester, Gene Tsudik, Spyros S. Magliveras, and Ivana Ilic, editors, *ISC*, volume 6531 of *Lecture Notes in Computer Science*, pages 99–113. Springer, 2010.

[SZ12] Frode Eika Sandnes and Xiaoli Zhang. User Identification Based on Touch Dynamics. In Bernady O. Apduhan, Ching-Hsien Hsu, Tadashi Dohi, Kenji Ishida, Laurence Tianruo Yang, and Jianhua Ma, editors, *UIC/ATC*, pages 256–263. IEEE Computer Society, 2012.

[UDWH13] Sebastian Uellenbeck, Markus Dürmuth, Christopher Wolf, and Thorsten Holz. Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM Conference on Computer and Communications Security*, pages 161–172. ACM, 2013.

[UW85] David A. Umphress and Glen Williams. Identity Verification Through Keyboard Characteristics. *International Journal of Man-Machine Studies*, 23(3):263–273, 1985.

[ZBHW12] Nan Zheng, Kun Bai, Hai Huang, and Haining Wang. You Are How You Touch: User Verification on Smartphones via Tapping Behaviour. Technical report, College of William & Mary, Williamsburg, VA, USA, December 2012.

[ZCPR03] Wen-Yi Zhao, Rama Chellappa, P. Jonathon Phillips, and Azriel Rosenfeld. Face Recognition: A Literature Survey. *ACM Comput. Surv.*, 35(4):399–458, 2003.

SNOOP-IT: Dynamische Analyse und Manipulation von Apple iOS Apps

Andreas Kurtz Markus Troßbach Felix C. Freiling

Department Informatik
Friedrich-Alexander-Universität
Martensstr. 3, 91054 Erlangen

{andreas.kurtz,felix.freiling}@cs.fau.de, markus@trossbach.net

Abstract: Applikationen für mobile Endgeräte (Apps) verarbeiten vielfach Daten, die unsere Privatsphäre betreffen. Bislang existieren aber kaum Werkzeuge, mit denen auf einfache Weise geprüft werden kann, wie Apps mit diesen Daten umgehen und ob dabei möglicherweise unsere Privatsphäre beeinträchtigt wird. Wir stellen das Werkzeug SNOOP-IT vor, mit dem bestehende Apps für die Apple iOS Plattform zur Laufzeit untersucht werden können. Am Beispiel einer populären App aus dem Apple App Store zeigen wir, wie unser Werkzeug dynamische Sicherheitsanalysen erleichtert und dabei unterstützt, Verletzungen der Privatsphäre effizient ausfindig zu machen.

1 Einleitung

In den vergangenen Jahren sind mobile Endgeräte wie Smartphones oder Tablet PCs ein fester Bestandteil unserer Lebenswelt geworden. Ein Grund für diesen Erfolg sind mitunter die vielfältigen Einsatzmöglichkeiten, die sich aus der Erweiterbarkeit durch Apps ergeben. Häufig sind sich App-Nutzer dabei allerdings der Risiken nicht bewusst, die sich beispielsweise aus dem Verlust ihres mobilen Endgeräts oder aus der unachtsamen Installation von Drittanbieter-Apps ergeben [BR13]. Dies gilt insbesondere für die Apple iOS Plattform, deren Quelltext im Gegensatz etwa zu Android, nicht öffentlich verfügbar ist. Insbesondere für diese Plattform existieren deshalb bislang kaum Werkzeuge, die es Sicherheitsfachleuten erlauben, iOS Apps effizient zu analysieren. Defizite innerhalb von Apps bleiben daher vielfach unentdeckt oder werden eher zufällig bzw. unter hohem Zeitaufwand gefunden [Tha, Ley, Bil].

Eine vergleichbare Ausgangssituation bestand bis vor wenigen Jahren auch bei der Analyse von Schadsoftware für Computer mit dem Windows-Betriebssystem. Dort haben sich aber mittlerweile dynamische Analysewerkzeuge etabliert, bei denen das zu untersuchende Programm unter kontrollierten Bedingungen ausgeführt wird und im Hintergrund sämtliche Aktivitäten, wie beispielsweise Dateisystem- oder Netzwerkzugriffe, zur Laufzeit überwacht werden [EGSF12, WHF07]. Obwohl sich die dynamische Analyse dabei als besonders effizient erwiesen hat, gibt es bislang kaum Werkzeuge zur Anwendung dieser Verfahren für mobile Apps.

1.1 Verwandte Arbeiten

In der Literatur gibt es verschiedene Arbeiten, die Ansätze zur Analyse mobiler Apps vorstellen. Die Forschung hat sich dabei in den vergangenen Jahren allerdings hauptsächlich mit der quelloffenen Android Plattform beschäftigt. Dort existieren Systeme wie Taintdroid [EGgC+10], Andrubis [Int13] oder Mobile Sandbox [SFE+13], die bereits jetzt eine automatisierte, dynamische Analyse von Android Apps erlauben.

Für Apples iOS Plattform ist das Angebot an Werkzeugen hingegen überschaubar. Neben ersten Ansätzen zur automatisierten dynamischen Analyse [SEKV12, JM12], existieren letztendlich kaum Werkzeuge, um Experten bei der zielgerichteten, manuellen Sicherheitsanalyse zu unterstützen. Die Werkzeuge Introspy [iSE] und iAuditor [MDS] verwenden API Hooking Techniken, um sicherheitsrelevante Vorgänge einer App aufzuzeichnen. Beide Werkzeuge beschränken sich dabei aber auf die Protokollierung grundlegender API-Aufrufe und bieten beispielsweise keine Möglichkeit zum Methoden-Tracing oder um mit der App zur Laufzeit zu interagieren. Darüber hinaus werden die von Introspy aufgezeichneten Daten nicht in Echtzeit ausgegeben und die von iAuditor registrierten API-Aufrufe werden lediglich in eine Datei protokolliert.

Das Werkzeug iNalyzer [Lab] beschränkt sich weitestgehend darauf, vorhandene App-Methoden aufzulisten und ermöglicht eine Ausführung dieser App-Methoden über eine Weboberfläche. Die Werkzeuge Aspective-C [Frea] und Subjective-C [Ken] ermöglichen ausschließlich eine Überwachung von Objective-C-Methodenaufrufe, beschränken sich dabei aber nicht auf App-eigene Methoden bzw. wichtige iOS API-Methoden, was ihre Nützlichkeit im Rahmen der App-Aanalyse einschränkt.

1.2 Resultate

In diesem Beitrag stellen wir das Werkzeug SNOOP-IT vor, das erste dynamische Analysewerkzeug zur zielgerichteten Analyse von Apps der Apple iOS Plattform, das Untersuchungsergebnisse in Echtzeit präsentiert. SNOOP-IT erweitert bestehende Apps mittels *Library Injection* zur Laufzeit um zusätzliche Funktionen zur Sicherheitsanalyse und kann somit Zugriffe einer App auf sicherheits- und privatsphärerelevante Methoden der iOS Plattform mittels *API Hooking* überwachen. Unser Werkzeug stellt dabei eine webbasierte Oberfläche zur Verfügung, um mit der überwachten App in Echtzeit zu interagieren. Durch eine XML-RPC-Schnittstelle ist es zudem möglich, automatisiert auf die aufgezeichneten Daten sowie interne App-Zustände zuzugreifen. Anhand der Analyse einer populären App zur Verwaltung von Passwörtern zeigen wir, wie SNOOP-IT zur zielgerichteten Analyse von sicherheitsrelevanter App-Funktionalität eingesetzt werden kann.

1.3 Ausblick

In Abschnitt 2 werden zunächst technische Hintergründe zur iOS Plattform und zum verwendeten API Hooking erläutert. Anschließend werden in Abschnitt 3 die einzelnen Komponenten von SNOOP-IT sowie deren Funktionsweise beschrieben. Im Rahmen der anschließenden Evaluation (Abschnitt 4) dokumentieren wir die Analyse der genannten App. Abschnitt 5 fasst die Ergebnisse zusammen.

SNOOP-IT ist unter der BSD Lizenz frei verfügbar und kann über die Projektseite `https://code.google.com/p/snoop-it/` heruntergeladen und ausprobiert werden.

2 Technische Hintergründe

In diesem Abschnitt werden technische Hintergründe zur iOS Plattform sowie Details zum eingesetzten API Hooking erläutert.

2.1 Objective-C

Alle Apps für das Apple iOS Betriebssystem werden in Objective-C entwickelt. Dabei handelt es sich um eine objektorientierte Erweiterung der Programmiersprache C. Darüber hinaus bietet Objective-C umfassende Möglichkeiten, um mit der zugrundeliegenden Ausführungsumgebung zu interagieren: Ein in Objective-C entwickeltes Programm kann beispielsweise sämtliche Werte und Methoden eigener Objekte zur Laufzeit einsehen oder auch verändern. Diese als Reflection bzw. Introspection bekannten Konzepte werden von der Objective-C Laufzeitumgebung zur Verfügung gestellt [Appb]. Die Funktionen der Laufzeitumgebung werden dabei beim Erstellen einer App über eine dedizierte Programm-Bibliothek (*libobjc.A.dylib*) zur App hinzu gelinkt.

Eine der wichtigsten Aufgaben dieser Laufzeitumgebung ist die Vermittlung von Nachrichten zwischen bestehenden Objekten (*Message Passing*). Beim Aufruf einer Methode wird eine Nachricht an das jeweilige Zielobjekt gesendet. Die Nachricht enthält dabei den Namen der aufzurufenden Methode sowie eine Liste der zugehörigen Parameterwerte. Um die Nachrichten zu vermitteln, stellt die Laufzeitumgebung über die Funktion `objc_msgSend` einen zentralen Dispatcher zur Verfügung. Der Aufruf einer App-Methode bzw. einer Methode der iOS-API hat folglich immer mindestens einen Aufruf dieses zentralen Dispatchers zur Folge.

2.2 Jailbreak

Mittels eines Jailbreaks ist es unter Ausnutzung von Software-Schwachstellen in der iOS-Plattform möglich, die Nutzungsbeschränkungen des iOS-Betriebssystems zu entfernen und administrative Rechte auf einem mobilen Endgerät zu erlangen. Darüber hinaus ermöglicht ein Jailbreak die Installation beliebiger Software (Analysewerkzeuge), die nicht von Apple signiert bzw. freigegeben wurde und bietet einen direkten Zugang zur Objective-C Laufzeitumgebung. Aus diesen Gründen erfolgt eine App-Analyse in der Praxis typischerweise auf Geräten mit durchgeführtem Jailbreak. Auch SNOOP-IT setzt zum Betrieb ein iOS-Gerät mit durchgeführtem Jailbreak voraus.

2.3 API Hooking

Um eine iOS App zur Laufzeit analysieren oder manipulieren zu können, muss zunächst der bestehende App-Programmcode um zusätzliche Funktionalität erweitert werden. Dies geschieht durch das Einbringen einer zusätzlichen Bibliothek (*Library Injection*). Dabei wird beim Start einer App der Dynamic Linker des iOS Betriebssystems über die Umgebungsvariable `DYLD_INSERT_LIBRARIES` angewiesen, den Code der angegebenen Bibliothek in den Adressraum der App zu laden. Anschließend muss bei der Initialisierung der Bibliothek sichergestellt werden, dass der eingeschleuste Programmcode auch tatsächlich aufgerufen wird (vgl. Abbildung 1). Dazu werden bestehende Methoden- und Funktionsaufrufe bzw. Zeiger auf die entsprechenden Codebereiche zur Laufzeit überschrieben, so dass anschließend anstatt der jeweiligen API Methode der zuvor über die Bibliothek eingeschleuste Code ausgeführt wird (*Runtime Patching*). Zur Unterstützung des API Hooking wird in SNOOP-IT das Mobile Substrate Framework verwendet [Freb].

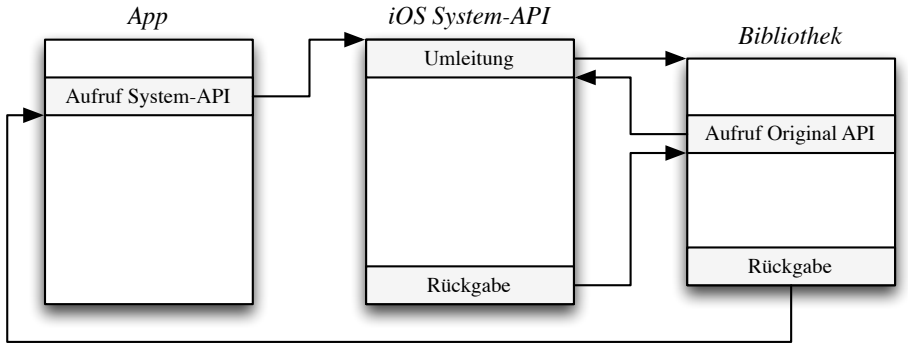


Abbildung 1: Überschreiben von iOS Methoden- bzw. Funktionsaufrufen zur Laufzeit mittels API Hooking.

3 Aufbau und Funktionsweise von SNOOP-IT

3.1 Grundprinzip

SNOOP-IT verwendet *Library Injection* und API Hooking zur Laufzeit, um zusätzliche Funktionalität zur Sicherheitsanalyse in eine iOS App einzubringen: Beim Start einer App wird diese durch das Einschleusen einer Bibliothek nachträglich um Analysefunktionen erweitert. Während der anschließenden Ausführung und Bedienung der App werden im Hintergrund zentrale sicherheits- und privatsphärerelevanten Vorgänge aufgezeichnet.

Um auf die aufgezeichneten Daten zuzugreifen und mit der App zu interagieren, wird innerhalb der App zusätzlich ein Webserver gestartet. Dieser Webserver stellt eine XML-RPC-Schnittstelle zur Verfügung, über die in Echtzeit auf die aufgezeichneten Ergebnisse und die Analysefunktionen zugegriffen werden kann. Zur einfachen Bedienung wird von dem Webserver zusätzlich eine webbasierte Benutzeroberfläche auf Basis des Google Web Toolkits [Sto12, gwt] bereitgestellt. Diese Benutzeroberfläche wiederum interagiert mittels AJAX mit der von der Bibliothek bereitgestellten Webservice-Schnittstelle und ermöglicht darüber einen einfachen Zugriff auf die SNOOP-IT-Funktionen.

Zur Auswahl der zu untersuchenden Apps stellt SNOOP-IT selbst eine eigene Konfigurations-App zur Verfügung. Darin werden sämtliche auf dem Analysegerät befindlichen Apps aufgelistet und können zur späteren Analyse markiert werden. Darüber hinaus können über die Konfigurations-App auch grundlegende Einstellungen wie beispielsweise der Netzwerkport des Webservers oder eine Authentifizierung zum Zugriff auf die Webservice-Schnittstelle vorgenommen werden.

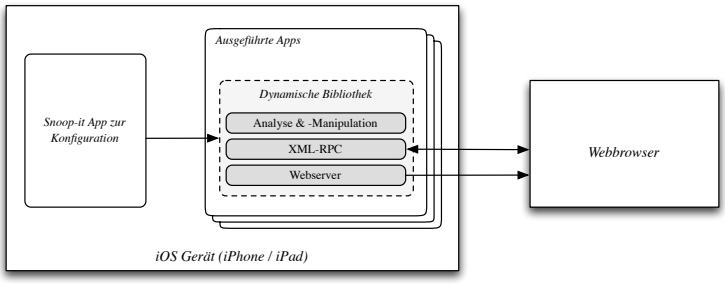


Abbildung 2: Durch das Einbringen einer dynamischen Bibliothek werden Apps zur Laufzeit um Funktionen zur Sicherheitsanalyse erweitert. Analyseergebnisse können in Echtzeit über einen Webbrowser eingesehen werden.

Abbildung 2 gibt einen Überblick über die in SNOOP-IT integrierten Komponenten: Nach dem Anpassen der Konfiguration, muss lediglich die jeweils zu untersuchende App gestartet und anschließend normal bedient werden. Während des Startvorgangs wird die SNOOP-IT-Programmbibliothek in die App integriert und die Überwachung gemäß der zuvor getroffenen Einstellungen begonnen. Der anschließende Zugriff auf die Analyseergebnisse

sowie die Interaktion mit der App zur Laufzeit erfolgt durch Aufruf der von SNOOP-IT bereitgestellten Weboberfläche im Browser.

Alternativ zum Zugriff über die Weboberfläche können die von dem Webservice bereitgestellten Funktionen auch von anderen Client-Anwendungen bzw. Werkzeugen direkt aufgerufen werden. Dazu findet sich auf der Projektseite von SNOOP-IT eine ausführliche Dokumentation der XML-RPC-Schnittstelle.

3.2 Bereitgestellte Funktionalität

In diesem Abschnitt werden die wichtigsten von SNOOP-IT bereitgestellten Funktionen erläutert.

3.2.1 Überwachung von API- und Systemaufrufen

Wird eine App um die Analysefunktionen erweitert, so werden anschließend zentrale App-Vorgänge überwacht und protokolliert. Dabei werden unter anderem sämtliche Zugriffe auf das Dateisystem und die iOS Keychain [Appa] registriert. In diesem Zusammenhang wird unter anderem auch ermittelt, ob eine App die vom iOS Betriebssystem bereitgestellten Verschlüsselungsmechanismen nutzt. Zahlreiche Filter innerhalb der grafischen Benutzeroberfläche erleichtern anschließend die Suche nach sensitiven App-Daten, die möglicherweise unverschlüsselt gespeichert werden und bei Verlust eines Geräts ein Risiko darstellen können.

Darüber hinaus überwacht SNOOP-IT den Netzwerkverkehr einer App sowie Aufrufe sämtlicher API-Methoden, über die eine App auf persönliche Informationen des Nutzers zugreifen kann (Kontakte, Fotos, Mikrofon, Kamera, Aufenthaltsort, Geräte-IDs etc.). Mit Hilfe dieser Überwachung kann relativ einfach nachvollzogen werden, ob eine App beispielsweise auf persönliche Daten oder sensitive Gerätesensoren zugreift.

3.2.2 Verfolgung des Kontrollflusses

Eine weitere Funktion in SNOOP-IT ermöglicht detaillierte Einblicke in den Kontrollfluss einer App. Dazu werden sämtliche Methodenaufrufe und zugehörige Parameterwerte protokolliert (*Tracing*). Auf diese Weise können beispielsweise interne Abläufe eingesehen werden, ohne dass dazu der eigentliche App-Quelltext benötigt wird.

Um Methodenaufrufe und zugehörige Parameterwerte aufzuzeichnen, werden sämtliche Nachrichten an den zentralen Dispatcher der Objective-C Laufzeitumgebung abgefangen. Mittels Hooking werden dabei Aufrufe der Dispatcher-Funktion `objc_msgSend` (siehe Abschnitt 2.1) überwacht. Dabei sind insbesondere die ARM CPU-Register $r0$ und $r1$ von Interesse, da diese Register beim Aufruf des Dispatchers jeweils einen Zeiger auf das Zielobjekt ($r0$) und einen Zeiger auf den Namen der auszuführenden Methode ($r1$) vorhalten. Da ein Protokollieren dieser Werte (beispielsweise in eine Datei) die Registerinhalte aber verändert würde, müssen die Zeiger vor dem Aufruf der Protokollierungsroutine

entsprechend gesichert werden. Dazu wird im Heap-Speicher eine Stack-Datenstruktur angelegt, um die original Registerwerte auf Instruktionsebene auf diesem alternativen Stack zu sichern. Nach der Protokollierung der Zeiger können die ursprünglichen CPU-Registerinhalte wiederhergestellt werden, bevor anschließend die normale Ausführung der App fortgesetzt wird.

Ein Nachteil dieses Ansatzes ist, dass dabei auch sämtliche Hintergrundaktivitäten der Laufzeitumgebung erfasst werden (beispielsweise Methodenaufrufe bei der Berührung der Displayoberfläche). Um dieses Funktionsrauschen auszublenden, wird in SNOOP-IT die Ausgabe des Tracings auf App-eigene Methoden und wichtige Funktionen der iOS Plattform beschränkt. Dazu werden zunächst über die Objective-C Laufzeitumgebung sämtliche von einer App bereitgestellten Klassen und Methoden ermittelt. Bei der Protokollierung der Nachrichtenaufrufe werden anschließend lediglich solche Aufrufe berücksichtigt, die von der App selbst oder in erster Instanz von der iOS API bereitgestellt werden.

3.2.3 App-Manipulation zur Laufzeit

Neben diesen Analysefunktionen stellt SNOOP-IT auch diverse Möglichkeiten bereit, um Apps zur Laufzeit zu manipulieren. So können über die Weboberfläche beispielsweise Hardware-Merkmale wie die Geräte-ID, die WLAN MAC-Adresse, der Gerätetyp oder der aktuelle Aufenthaltsort einfach vorgetäuscht werden. Darüber hinaus ermittelt SNOOP-IT aus dem Programmcode einer App alle verfügbaren Klassen und Methodennamen und stellt diese als Baumstruktur dar. Dabei werden auch Initialisierungen von Klassen überwacht und jeweils eine Referenz auf die neu angelegten Instanzen zwischengespeichert. Dadurch können anschließend vorhandene Instanzen bzw. Klassen über die Weboberfläche ausgewählt und deren Methoden beliebig aufgerufen werden.

4 Evaluation

Die Möglichkeiten, die sich aus der zielgerichteten, dynamischen Analyse von Apps mittels SNOOP-IT ergeben, sind vielfältig. So kann zur Laufzeit beispielsweise sehr einfach festgestellt werden, auf welche persönliche Daten oder sensitive Gerätesensoren eine App zugreift. Diese Informationen können anschließend verwendet werden, um daraus wiederum potentielle Privatsphäreverletzungen abzuleiten. Im Gegensatz zu automatisierten Ansätzen [SEKV12, SFE+13] versucht SNOOP-IT dabei nicht, schadhaftes Verhalten automatisiert aufzudecken und zu bewerten. Vielmehr ermöglicht es unser Werkzeug einem Experten auf sehr effiziente Weise, interne Abläufe einer App zu untersuchen und daraus Rückschlüsse auf mögliche Schadfunktionen zu ziehen.

Ferner unterstützt SNOOP-IT dabei, technische Defizite innerhalb der Implementierung einer App aufzudecken, die ihrerseits wiederum Auswirkungen auf die Privatsphäre haben können. Um beispielsweise Integrität und Vertraulichkeit lokal gespeicherter Daten auch bei Verlust eines mobilen Endgeräts weiterhin gewährleisten zu können, werden sensitive App-Inhalte meist verschlüsselt. In diesem Zusammenhang wird bei Prüfungen häufig der

Fragestellung nachgegangen, ob die hinterlegten Daten durch das eingesetzte Verschlüsselungsverfahren effektiv geschützt werden. Am Beispiel einer populären App zur Verwaltung von PINs und Passwörtern demonstrieren wir im Folgenden, wie SNOOP-IT dazu verwendet werden kann, solchen Fragestellungen nachzugehen und sicherheitsrelevante Implementierungsfehler effizient aufzudecken.

Fallbeispiel: App zur sicheren PIN- und Passwortverwaltung

Die im Apple App Store verfügbare App *iPIN* wirbt damit, sämtliche "Daten (..) mit dem Advanced Encryption Standard und einer Schlüssellänge von 256 Bit" zu verschlüsseln [IBI]. Der Zugriff auf die verschlüsselten Daten soll erst dann möglich werden, wenn über die integrierte "Sensor-Tastatur" ein zuvor aufgezeichnetes Entsperrmuster eingegeben wurde.

Durch die in SNOOP-IT integrierte Dateisystemüberwachung (vgl. Abbildung 3) wurde festgestellt, dass die App direkt beim Start auf die Datei `iPinSecurityModel.dat` zugreift. Diese Datei wird nicht über die Verschlüsselungs-API des iOS Betriebssystems geschützt (*NSFileProtectionNone*), weshalb die darin enthaltenen Daten bei Verlust eines Geräts ausgelesen werden könnten. Ein Zugriff auf die Datei zeigte allerdings, dass deren Inhalte binär bzw. kodiert vorliegen und nicht unmittelbar einsehbar sind.

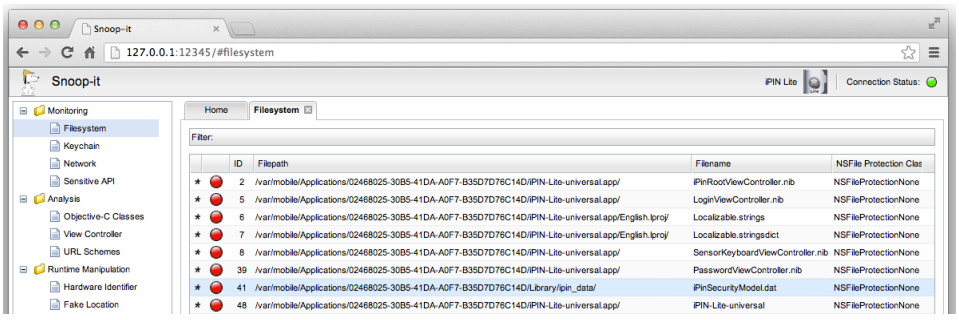


Abbildung 3: Überwachung von Dateisystemzugriffen

Mit Hilfe der in SNOOP-IT integrierten Funktion zur Verfolgung von Methodenaufrufen wurde anschließend untersucht, wie diese Datei nach dem Einlesen weiterverarbeitet wird. Listing 1 zeigt, dass direkt nach dem Einlesen der Datei mittels Key Stretching (*PBKDF2*) aus einer im App-Programmcode statisch hinterlegten Zeichenkette ein Schlüssel abgeleitet wird (vgl. Zeile 3). Unter Verwendung dieses Schlüssels werden die Daten der Datei anschließend entschlüsselt und die entschlüsselten Werte im Objekt *iPINModel* zwischengespeichert (vgl. Zeile 4). Wie sich der Ausgabe entnehmen lässt, befindet sich in der Datei unter anderem der MD5-Hashwert des erwarteten Entsperrmusters (vgl. *sensorHash* in Zeile 7).

```
1 + [iPinModel(0x90f68) initFromFile]
2 + [iPinModel(0x90f68) securityModelFilePath]
3 + [PBKDF2(0x9124c) getKeyForPassphrase:], args: <__NSCFConstantString 0x92160:
      [initForWritingWithMutableData]>
4 + [iPinModel(0x90f68) initSharedModelWithUnarchiver:withObjectKey:], args: <0
      x2002aef0>, <__NSCFConstantString 0x92150: iPINModel>
5 + [iPinModel(0x90f68) sharedModel]
6 - [iPinModel(0x200e2130) initWithCoder:], args: <0x2002aef0>
7 - [iPinModel(0x200e2130) setSensorHash:], args: <__NSCFString 0x2002a630:
      8CF37F50FB1A7943FBA8EAA20FFF1E56>
```

Listing 1: Methodenaufrufe beim Start der App zeigen die Verwendung eines im Programmcode der App statisch hinterlegten Schlüssels

Weitere Analysen mithilfe der Tracing-Funktion ergaben, dass die von der Sensor-Tastatur bereitgestellten Tasten durchnummeriert sind (vgl. Abbildung 4). Bei jedem Tastendruck wird der aktuelle Wert mit den zuvor gedrückten Tastenwerten zusammengeführt und davon ein MD5-Hashwert errechnet. Dieser wird mit dem zuvor aus der Datei ermittelten Hashwert (*sensorHash*) verglichen.

Da das erwartete Entsperrmuster in Form des MD5-Hashwerts auf dem Gerät selbst gespeichert wird und der zugehörige Wertebereich relativ klein ist, werden dadurch Brute-Force-Angriffe begünstigt. Ein beliebiges Muster bestehend aus neun gedrückten Sensortasten ließ sich in Praxistests über ein Python-Script in nur wenigen Sekunden ermitteln. Der eingegebene Sensor-Code wird von der App anschließend verwendet, um daraus erneut einen Schlüssel abzuleiten und damit die eigentlichen App-Daten zu entschlüsseln.

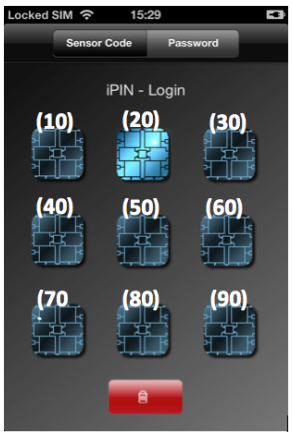


Abbildung 4: Sensor-Tastatur zur Eingabe eines Entsperrmusters

Da durch diesen Implementierungsfehler die Authentifizierung und App-eigene Verschlüsselung umgangen werden kann, können Integrität und Vertraulichkeit der durch die App verarbeiten Daten bei Verlust eines mobilen Endgeräts nicht sichergestellt werden. Direkt nach dem Entdecken der Schwachstellen wurde daher der Hersteller der App darüber informiert. Seit Anfang Juli 2013 steht eine aktualisierte Version der App zur Verfügung, in der die Befunde durch Überarbeitung des Verschlüsselungsverfahrens adressiert wurden.

5 Zusammenfassung

Die dynamische Analyse von iOS Apps zur Laufzeit gewährt detaillierte Einblicke in deren Innenleben. Verletzungen der Privatsphäre können dabei häufig ebenso einfach identifiziert werden, wie sicherheitsrelevante Implementierungsfehler. Mit SNOOP-IT stellen wir ein einfach zu bedienendes Werkzeug zur Verfügung, um solche Defizite zukünftig effizienter ermitteln zu können. Neben diesen technischen Analysen, ermöglicht die Interaktion zur Laufzeit aber auch völlig neue Angriffswege und zwar immer dann, wenn Sicherheitsmaßnahmen innerhalb einer App umgesetzt werden. Apps sollten daher verstärkt unter dem Bewusstsein entwickelt werden, dass sicherheitsrelevante Vorgänge auf Client-seite (wie statische Verschlüsselungsschlüssel, hartkodierte Zugangsdaten, vorgelagerte Anmeldemasken etc.) beliebig eingesehen, manipuliert und umgangen werden können.

Literatur

[Appa] Apple. Keychain Services Programming Guide. `https://developer.apple.com/library/ios/documentation/security/conceptual/keychainServConcepts/01introduction/introduction.html`.

[Appb] Apple. Objective-C Runtime Reference. `https://developer.apple.com/library/mac/documentation/cocoa/reference/objcruntimeref/Reference/reference.html`.

[Bil] Nick Bilton. Apple Loophole Gives Developers Access to Photos. `http://bits.blogs.nytimes.com/2012/02/28/tk-ios-gives-developers-access-to-photos-videos-location/`.

[BR13] Zinaida Benenson und Lena Reinfelder. Should the Users be Informed? On Differences in Risk Perception between Android and iPhone Users. In *Symposium on Usable Privacy and Security (SOUPS)*, 2013.

[EGgC+10] William Enck, Peter Gilbert, Byung gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel und Anmol Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In Remzi H. Arpaci-Dusseau und Brad Chen, Hrsg., *9th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2010, October 4-6, 2010, Vancouver, BC, Canada, Proceedings*, Seiten 393–407. USENIX Association, 2010.

[EGSF12] Markus Engelberth, Jan Göbel, Christian Schönbein und Felix C Freiling. PyBox-A Python Sandbox. In *Sicherheit*, Seiten 137–148, 2012.

[Frea] Jay Freeman. Aspective-C. `http://svn.saurik.com/repos/menes/trunk/aspectivec/AspectiveC.mm`.

[Freb] Jay Freeman. Mobile Substrate. `http://www.cydiasubstrate.com/`.

[gwt] Google Web Toolkit. `http://www.gwtproject.org/`.

[IBI] IBILITIES, INC. iPIN - Secure PIN & Passwort Safe App. `https://itunes.apple.com/de/app/ipin-secure-pin-passwort-safe/id379865087`.

[Int13] International Secure Systems Lab. Anubis: Analyzing Unknown Binaries. online http://anubis.iseclab.org/, November 2013.

[iSE] iSECPartners. Introspy. https://github.com/iSECPartners/introspy/.

[JM12] Mona Erfani Joorabchi und Ali Mesbah. Reverse engineering iOS mobile applications. In *Reverse Engineering (WCRE), 2012 19th Working Conference on*, Seiten 177–186. IEEE, 2012.

[Ken] KennyTM~ Subjective-C. http://networkpx.blogspot.de/2009/09/introducing-subjective-c.html.

[Lab] AppSec Labs. iNalyzer. https://appsec-labs.com/iNalyzer.

[Ley] John Leyden. D'OH! Use Tumblr on iPhone or iPad, give your password to the WORLD. http://www.theregister.co.uk/2013/07/17/tumblr_ios_uncryption/.

[MDS] MDSec. iAuditor. https://github.com/mdsecresearch.

[SEKV12] Martin Szydlowski, Manuel Egele, Christopher Kruegel und Giovanni Vigna. Challenges for dynamic analysis of iOS applications. In *Open Problems in Network Security*, Seiten 65–77. Springer, 2012.

[SFE+13] Michael Spreitzenbarth, Felix C. Freiling, Florian Echtler, Thomas Schreck und Johannes Hoffmann. Mobile-sandbox: having a deeper look into android applications. In Sung Y. Shin und José Carlos Maldonado, Hrsg., *SAC*, Seiten 1808–1815. ACM, 2013.

[Sto12] Sebastian Stocker. Entwurf einer grafischen Oberfläche zur Laufzeitmanipulation von iOS Apps. Bachelor Thesis, Universität Heidelberg, Hochschule Heilbronn, Medizinische Informatik, 2012.

[Tha] Arun Thampi. Path uploads your entire iPhone address book to its servers. http://mclov.in/2012/02/08/path-uploads-your-entire-address-book-to-their-servers.html.

[WHF07] Carsten Willems, Thorsten Holz und Felix Freiling. Toward automated dynamic malware analysis using cwsandbox. *Security & Privacy, IEEE*, 5(2):32–39, 2007.

Sämtliche angegebenen Online-Quellen wurden zuletzt am 26.01.2014 erfolgreich abgerufen.

GraphNeighbors: Hampering Shoulder-Surfing Attacks on Smartphones

Irfan Altiok, Sebastian Uellenbeck, Thorsten Holz

Horst Görtz Institute for IT-Security
Ruhr-University Bochum
Universitätsstrasse 150
44810 Bochum
irfan.altiok@rub.de
sebastian.uellenbeck@rub.de
thorsten.holz@rub.de

Abstract: Today, smartphones are widely used and they already have a growing market share of more than 70 % according to recent studies. These devices often contain sensitive data like contacts, pictures, or even passwords that can easily be accessed by an attacker if the phone is not locked. Since they are mobile and used as everyday gadgets, they are susceptible to get lost or stolen. Hence, access control mechanisms such as user authentication are required to prevent the data from being accessed by an attacker. However, commonly used authentication mechanisms like PINs, passwords, and Android Unlock Patterns suffer from the same weakness: they are all vulnerable against different kinds of attacks, most notably shoulder-surfing. A promising strategy to prevent shoulder-surfing is to only enter a derivation of the secret during the authentication phase.

In this paper, we present a novel authentication mechanism based on the concept of *graphical neighbors* to hamper shoulder-surfing attacks. Results of a usability evaluation with 100 participants show that our implementation called GRAPHNEIGHBORS is applicable in comparison to commonly used authentication mechanisms.

1 Introduction

Smartphones are among the most popular gadgets available on the market today. According to recent studies, their popularity is expected to grow even more in the near future. Since smartphones are not only used for calling but also for taking pictures, sending text messages, surfing the web, or online banking, they contain a lot of sensitive and private data like contact information, personal messages, or even passwords. Due to their mobility, they are carried around by users and consequently, they are an interesting target for attackers, who can easily access all sensitive data if the smartphone is not locked. Hence, smartphones offer different authentication mechanisms like passwords, PINs, or *Android Unlock Patterns*. Since passwords and PINs are cumbersome to enter into the device, alternative solutions are needed. Several studies showed that visual information like pictures are easier to recall than textual information [SCH70] and thus Android Un-

lock Patterns seem to be the solution. However, it also has been shown that the entropy of user-chosen Android Unlock Patterns is rather low [UDWH13]. In the same way, all three methods suffer from the same weakness: they are vulnerable against shoulder-surfing attacks [TOH06]. In this paper, we present a concept and an implementation of GRAPHNEIGHBORS, that counters plain shoulder-surfing attacks while serving usability as well.

Ethical Considerations

As part of our work, we asked 100 people to authenticate using the developed prototypes. All users were informed before participating that they were to take part in a scientific study and that their data was used to analyze the properties of an authentication system. All data collected during our study was used in an anonymized way so that there is no link between collected data and individual participants. Our institute does not fall under the jurisdiction of an IRB or similar ethics committee.

Paper Outline

The rest of this paper is structured as follows. In Section 2, we discuss related work in particular on shoulder-surfing resilient user authentication. We explain our approach in detail and discuss our prototype implementation in Section 3. In Section 4, we present the results of a user survey with 100 participants with focus on the usability of GRAPHNEIGHBORS. We introduce four attacker scenarios and argue how GRAPHNEIGHBORS performs in contrast to legacy authentication approaches like PIN, password, and Android Unlock Patterns in Section 5. Finally, we conclude this paper in Section 6 with a discussion of the results and potential future research questions.

2 Related Work

In recent years, a lot of work has been published about user authentication on mobile devices but only a few aims at hampering shoulder-surfing attacks.

Tan et al. proposed a spy-resistant keyboard that should allow for more secure password entry on public touch screens [TKC05]. Their keyboard is composed of 42 character tiles, two inter-actor tiles, a feedback text box, a backspace button, and an enter button. Each tile randomly contains a lower letter, an upper letter, and a digit or a symbol. Instead of having a fixed shift state, each tile has a random assigned shift state defined by a red line under one of the three characters. By pressing one of the inter-actor tiles, the user can change the shift state of all tiles simultaneously. Having located the correct character as well as the appropriate shift state, he has to drag an inter-actor tile on the character tile to select it. While dragging to the correct tile, all shift states disappear. Balzarotti et al. showed that this schema can easily be broken by a single shoulder-surfing session since

the password is always entered in clear [BCV08]. GRAPHNEIGHBORS improves on this since the login session can be observed several times without revealing the secret.

Roth et al. presented a PIN entry method to mitigate shoulder surfing [RRF04]. On a keypad, they colored the numbers into black and white groups and the user has to select the group his PIN's digit is part of in a round-based fashion. A shoulder surfer would only obtain a subset of the potential PIN, but she could reconstruct the PIN with an intersection analysis if she would be able to watch the entry process several times. In contrast to our approach, the authors assume that the attacker's resources are bounded by the cognitive capabilities of a human. Again, this only holds if the attacker is not allowed to take notes or record the authentication procedure, for example with a (smartphone) camera [BCV08, MVG+11]. Otherwise, she could intersect the chosen subsets to obtain the secret PIN within a few rounds. GRAPHNEIGHBORS resists such attacks.

S3PAS is a system implemented by Zhao and Li to prevent shoulder-surfing attacks [ZL07]. Here the secret is a list of three characters where each item forms a virtual and invisible triangle in a 10×10 grid of characters. To authenticate, the user has to touch inside the triangle in a row-based fashion. Again, this scheme is not secure against sophisticated shoulder-surfing attacks where the attacker records a few sessions [BCV08, MVG+11]. By intersecting the possible password candidates she can afterwards obtain the secret.

3 Methodology and Implementation

Currently, Android offers four mechanisms to unlock a smartphone. The most secure in terms of theoretical password space but also least usable method are passwords. The user can choose an arbitrary string containing digits (0-9), letters (a-zA-Z), and special characters (e. g., !,?,;,:) to form a login secret on a hardware or a software keyboard. There is only one constraint: The login secret has to contain at least 4 characters and at most 16. Although this results in a total password space of about 2^{107}, most users do not want to utilize such a cumbersome approach since entering a secret on a softkeyboard is error-prone due to small buttons and also takes some time.

The PIN login mechanism is a special case of the aforementioned passwords. Here, the user has a limited character set of only digits with at least 4 and at most 16 digits that leads to approximately 2^{53} different secrets. Since the buttons to enter the secret are much larger in comparison to the softkeyboard used for entering password, this authentication method is faster and less error-prone.

The third alternative are Android Unlock Patterns, a simplified variant of the Pass-Go scheme [TA08] to increase usability and to adapt for the small screens found on typical devices running Android. It uses nine points arranged in a 3×3 grid and the user needs to select a path through these points according to the following rules:

1. At least four points must be chosen,
2. no point can be used twice,

3. only straight lines are allowed, and
4. one cannot jump over points not visited before.

Since the authentication can be performed very quickly, this method is a powerful replacement for password and PIN. However, all three mechanisms suffer from the very same weakness: they are susceptible to shoulder-surfing attacks. Besides this, Aviv et al. have shown that the secret can often be obtained by an attacker through analyzing the oily residue left by the user's touch [AGM+10] (so called *smudge attacks*).

The last authentication mechanism is Android's Face Login. To make use of this approach, the user has to take a picture of his face and afterwards look into the camera in order to authenticate. On the one hand, this leads to resistance against classic shoulder-surfing since the secret cannot be easily eavesdropped by an attacker. On the other hand, it has been shown that an attacker only needs a picture of the smartphone's owner to authenticate. A countermeasure has been implemented in Android 4.1 where the user has to blink in order to login. Albeit, we found that even this can easily be fooled by using two pictures (one with open eyes and another one with closed eyes).

In summary, Face Login is not secure against simple picture attacks whereas password, PIN, and Android Unlock Patterns suffer from shoulder-surfing attacks. Thus there is a need for an authentication method that solves these shortcomings. Multiple extensions for the PIN system have been proposed in recent years [DLFBH09, DLHH10, DLWD07, BOKK11, RRF04] that attempt to hamper shoulder-surfing. While recalling visual information like pictures is easier than recalling passwords or PINs [SCH70, BCVO12, Chi08], there are only a few approaches that make use of graphical secrets that cover this attack vector [FCB10].

We propose GRAPHNEIGHBORS, a graphical authentication mechanism that fixes these drawbacks. To evaluate different settings, we develop three approaches as prototypes that we describe in the following. We show that plain shoulder-surfing attacks do not reveal the login secret and additionally that smudge attacks are useless against our scheme. Like other authentication mechanisms, GRAPHNEIGHBORS utilizes different figures as secrets. We denote this set of figures as $\mathbb{F}$. Additionally, GRAPHNEIGHBORS uses a set of colors $\mathbb{C}$ as well as a set of optional positions $\mathbb{P}$. The user's secret is based on figures having each figure $f_i \in \mathbb{F}$ being concatenated with a user chosen color $c_i \in \mathbb{C}$ and a user chosen position $p_i \in \mathbb{P}$. The position refers to the selection the user has to choose when authenticating. In contrast to most existing graphical logins, the user does not select his secret directly to authenticate. Instead, he chooses the figure *besides* his partial secret to prevent shoulder-surfing attacks. The previously selected position is the key to decide which of the neighbors to choose. Therefore, a four-digit user's secret $\mathbb{S} = (s_1, s_2, s_3, s_4)$ can be described as the following ordered set:

$$(f_1 \circ c_1 \circ p_1, f_2 \circ c_2 \circ p_2, f_3 \circ c_3 \circ p_3, f_4 \circ c_4 \circ p_4)$$

with $s_i = f_i \circ c_i \circ p_i$. Note that $\circ$ denotes the concatenation operator. To configure the login, the user has to choose a secret $\mathbb{S}$ beforehand.

As a first approach, we choose six different geometric figures (`circle` ◯, `square` □, `triangle` △, `rhombus` ◊, `pentagon` ⬠, and `star` ★), four different colors (`blue`,

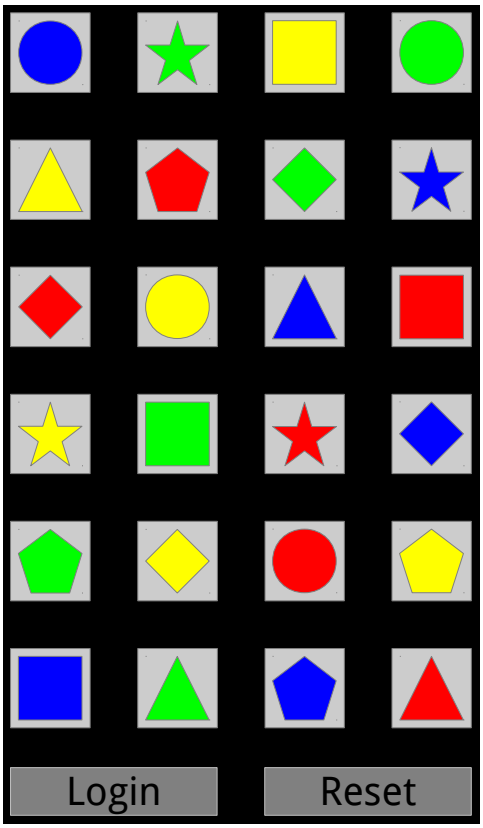


Figure 1: View for approach A and B. The user has to choose at least four objects as secret. In contrast to approach B, approach A additionally uses directions as secret for each object.

`green`, `yellow`, and `red`), and five different locations (`above`, `right`, `below`, `left`, and `none`). The login view for this approach can be seen in Figure 1 presenting a matrix of 6×4 figures randomly located. The authentication process is round-based: For each round, the user has to find the figure that matches his chosen secret regarding shape and color and select the figure corresponding to his position selection beside the located figure. If $f_i \circ c_i$ is located at the upper left edge of the screen and the previously chosen position p_i is left, he has to choose the most right figure in the same row. If p_i is above, the user has to select the bottommost figure in the same column. Other cases match accordingly. Like for other methods that have a user defined secret's length, the user has to press a button after having inserted the full secret $\mathbb{S}$.

To give a concrete example, we explain the authentication process based on Figure 1. Let the first partial secret be $f_1 \circ c_1 \circ p_1$ with $f_1 = \bigcirc$, $c_1 =$ `red`, and $p_1 =$ `below`. Given the aforementioned figure, the user has to select the `blue` $\pentagon$ since this figure is right below the red circle. Let the second partial secret be $f_2 \circ c_2 \circ p_2$ with $f_2 = \square$, $c_2 =$ `blue`, and $p_2 =$ `left`. For the second round, the user has to select the red triangle since the square

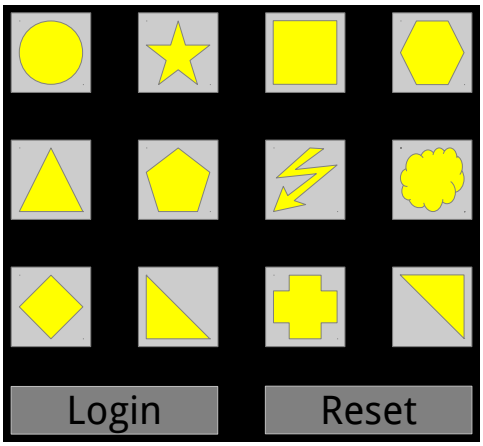


Figure 2: View for approach C. The user has to choose at least four objects as secret. Additionally, directions for each object must be chosen.

is the leftmost figure in the last row.

Our second approach is very similar to the first one: We also choose six different geometric figures and four colors, but the user cannot choose different locations on his own. Thus, the automatic selection for this property is `none`. Since we omit the neighbor option, this method is not secure against shoulder-surfing attacks. By offering this approach, we intended to evaluate whether more difficult approaches like approach A are usable in comparison to omitting this feature.

Our final approach differs from the first and second one by the number of figures and the number of colors (c. f. Figure 2). We implemented twelve different figures (`circle` ◯, `star` ★, `square` □, `hexagon` ⬡, upward directed isosceles `triangle` △, `cloud` ☁, `pentagon` ⬠, `lightning` 🗲, `rhombus` ◊, `cross` ✜, left downward directed isosceles `triangle` ◺, and right upward directs isosceles `triangle` ◹) with five different locations as in approach A, but used only a single color. We considered this approach since we wanted to find out whether the concept of neighbors works with only one color but more figures. On a smartphone, these figures can easily be taken from the user's gallery since there is no need to colorize them. Note that this is only feasible for approach C since some pictures might be indistinguishable when being colored as needed for the approaches A and B. For our prototype, we used geometric figures due to the easy comparability. Pictures instead of figures might be more familiar for the smartphone's owner, but less comparable for a study.

We implemented all three approaches as prototypes in a tool called GRAPHNEIGHBORS for the Android operating system. A summary of the method's properties can be found in Table 1. In the next section, we evaluate and discuss the usability of them.

Table 1: Comparison of the properties of different approaches.

	Figures	Colors	Neighbors	Displayed Objects
Approach A	6	4	5	24
Approach B	6	4	1	24
Approach C	12	1	5	12

4 Usability Evaluation

To evaluate GRAPHNEIGHBORS, we conducted a user study with 100 participants. In the beginning, we explained the GRAPHNEIGHBORS to each participant face to face and answered upcoming questions. We further asked them to select and remember a secret for each approach individually. After five minutes, we requested each participant to login with the previously chosen secrets having three tries for each approach. Finally, we asked them to fill out a questionnaire.

We learned that $74\,\%$ of all participants where able to authenticate using approach A within three attempts. $86\,\%$ of all participants could login when using approach B while $96\,\%$ could authenticate by means of approach C. Table 2 outlines details about successful logins and the number of attempts participants needed to authenticate. By intersecting the sets of successful participants we discovered that $68\,\%$ could authenticate using each single approach and that almost all subjects ($98\,\%$) could authenticate using at least one of the approaches.

We further asked our participants to compare all three approaches of GRAPHNEIGHBORS with common login methods and name the method where they could remember the secret best. $27\,\%$ stated that they could remember secrets for approach C best, while $25\,\%$ could remember PINs best. $18\,\%$ liked passwords most followed by $15\,\%$ preferring Android Unlock Patterns. Approach A was chosen by only $11\,\%$ and approach B by only $4\,\%$. Note that approach C performs better than all legacy authentication methods while being secure against simple shoulder-surfing attacks (c. f. Section Security Evaluation).

We also asked which authentication method they would prefer as default for their smartphone. $34\,\%$ gave approach C as preferred method followed by PIN with $19\,\%$ and approach A as well as other mechanisms with $14\,\%$ each. Passwords achieved $10\,\%$ while approach B got $5\,\%$ and Android Unlock Patterns $4\,\%$.

Table 2: Number of attempts to successfully login for 100 participants.

	Approach A	Approach B	Approach C
1^{st} try	42	59	94
2^{nd} try	28	25	2
3^{rd} try	4	2	-
Σ	74	86	96

5 Security Evaluation

In this section, we introduce four attacker scenarios and argue how GRAPHNEIGHBORS performs in contrast to legacy authentication approaches like PIN, password, and Android Unlock Pattern. We concentrate on the average case since worst or best case scenarios do not point out reality.

The first attack we discuss is guessing the secret without knowing the secret's length. Since $\sum_{i=4}^{n} b^i$ is the password space for passwords with at least 4 and at most n characters having a basis of b different characters, we need on average $\frac{\sum_{i=4}^{n} b^i}{2}$ tries to reveal a password. Given that, we get $\frac{\sum_{i=4}^{16} 95^i}{2} \approx 1.09 \times 2^{104}$ as concrete number. Transferred to the PIN authentication, we obtain $\frac{\sum_{i=4}^{16} 10^i}{2} \approx 1.23 \times 2^{52}$ for $b = \{0, \ldots, 9\}$. For Android Unlock Patterns an exact number cannot be given in a closed equation since login patterns have a lot of restrictions, e. g., points cannot be used twice and not every path is allowed. Considering all restrictions, we nearly get 400.000 possible patterns. Therefore, we need on average $200.000 \approx 1.53 \times 2^{17}$ tries to expose the login secret. Comparing this to approach A of GRAPHNEIGHBORS, we have a base of $6 \times 4 = 24$ different object/color combinations. Although we additionally can choose five different directions, this does not change the security of this approach because all possible objects are already element of these 24 objects. Hence, we obtain $\frac{\sum_{i=4}^{16} 24^i}{2} \approx 1.34 \times 2^{72}$ as average count of attacks to guess the secret. For approach B, we get the very same number since we do not need to take the number of neighbors into account. This differs for approach C where we have only 12 different objects. Given this, we get $\frac{\sum_{i=4}^{16} 12^i}{2} \approx 1.40 \times 2^{56}$ as average number of attacks to guess the secret.

As second attack, we consider guessing the secret with knowledge of it's length. For our analysis, we assume a secret's length of four. Taking a look at passwords, we obtain $95^4 \approx 1.21 \times 2^{26}$ different possibilities. Therefore, an attacker needs approximately 2^{25} tries to expose the secret. For four digit PINs, we get $\frac{10^4}{2} \approx 1.22 \times 2^{12}$ possibilities. Android Unlock Patterns containing exactly four points only offer less than $\frac{9 \times 8 \times 7 \times 6}{2} \approx 1.47 \times 2^{10}$ since the first point can be chosen arbitrarily but further points depend on the previously chosen ones. Calculating this for GRAPHNEIGHBORS, we obtain $\frac{24^4}{2} \approx 1.27 \times 2^{17}$ for approach A and B, while we need on average $\frac{12^4}{4} \approx 1.27 \times 2^{14}$ tries to get the secret for approach C.

The third attack is guessing the user's secret after shoulder-surfing. When considering password, PIN, and Android Unlock Patterns, we conclude that all these approaches can easily be broken by conducting shoulder-surfing once. An attacker having observed the login procedure knows the secret and can unlock the smartphone afterwards. Since approach B of GRAPHNEIGHBORS does not make use of the neighbor option, this method can also be attacked by simple shoulder-surfing. However, approach A and C make use of the neighbor option and are therefore secure against single shoulder-surfing attacks since the attacker does not attain the secret. Albeit, they are not resilient against sophisticated shoulder-surfing attacks because an attacker conducting multiple shoulder-surfing sessions can perform an intersection attack to obtain the secret in the end. To do so, the

attacker captures the user's inputs including the possible neighbors and creates a list of password candidates. Having four rounds and five possibilities per round, the list consists of $5^4 = 625$ candidates. Due to the randomization of the figures on the touchscreen, each login session offers other candidates. By collecting and intersecting multiple candidate sets, the attacker can eventually obtain the secret.

Finally, we consider smudge attacks [AGM+10]. In this context, the oily residue on the smartphone's touchscreen is analyzed to obtain the login secret. Here, an attacker can inspect the touchscreen after the user has tapped or wiped on the touchscreen to enter the secret. The Android pattern login is most vulnerable against this attack vector. Since the user has to slide his finger to enter the secret, residues form a path on the touchscreen. This path has exactly two designated points: on the one hand the start point and on the other hand the end point. Start and end point can be interchanged but this only leads to two possible secrets. Having five tries to authenticate using this method, we can consider that plain login patterns are highly vulnerable. By looking at password and PIN, attackers can see where a user has touched the touchscreen. Therefore, they know the touched position but have no information about the order of touches. For a secret with four different characters (numbers or letters) we get $4! = 24$ possible secrets leading to 1.5×2^3 tries on average. Since all approaches of GRAPHNEIGHBORS show the geometric figures randomly arranged, smudge attacks do not lead to any hint of the secret.

Limitations

While being more secure against shoulder-surfing attacks than PIN, password, and Android Unlock Patterns, GRAPHNEIGHBORS also has at least one limitation. Since the number of selectable neighbors is limited, observing the authentication reveals a hint of the login secret. By intersecting the possible password candidates, this can lead in the long run to the secret password. This attack can only be mitigated by changing the password on a regular basis. Nevertheless, GRAPHNEIGHBORS hampers shoulder-surfing attacks since even a sophisticated shoulder-surfing attack performed once does not lead to the login secret.

6 Conclusion and Future Work

In this paper, we introduced a novel graphical authentication method that makes use of neighbors to achieve resistance against plain shoulder-surfing attacks. We implemented three different approaches as a prototype for the Android OS. In a usability evaluation, we asked 100 people to choose secrets for each approach and verified that almost all participants ($98\,\%$) where able to remember the secret after a short period of time of at least one approach. In a succeeding survey we found that $34\,\%$ preferred using our proposed neighbor option with 12 different figures but without additional colors. Our security evaluation shows that graphical neighbors improve the security against shoulder-surfing on the

one hand and smudge attacks on the other compared to commonly used authentication methods.

In future work, we plan to focus on a large-scale and long-term evaluation to verify that people can remember not only figures, but also positions over a long period of time.

References

[AGM+10] Adam J. Aviv, Katherine Gibson, Evan Mossop, Matt Blaze, and Jonathan M. Smith. Smudge Attacks on Smartphone Touch Screens. In *USENIX Workshop on Offensive Technologies (WOOT)*, 2010.

[BCV08] Davide Balzarotti, Marco Cova, and Giovanni Vigna. ClearShot: Eavesdropping on Keyboard Input from Video. In *IEEE Symposium on Security and Privacy*, pages 170–183. IEEE Computer Society, 2008.

[BCVO12] Robert Biddle, Sonia Chiasson, and P.C. Van Oorschot. Graphical Passwords: Learning From the First Twelve Years. *ACM Computing Surveys*, 44(4):19:1–19:41, September 2012.

[BOKK11] Andrea Bianchi, Ian Oakley, Vassilis Kostakos, and Dong-Soo Kwon. The Phone Lock: Audio and Haptic Shoulder-Surfing Resistant PIN Entry Methods for Mobile Devices. In *Tangible and Embedded Interaction*, 2011.

[Chi08] Sonia Chiasson. *Usable Authentication and Click-based Graphical Passwords*. PhD thesis, Carleton University, 2008.

[DLFBH09] Alexander De Luca, Bernhard Frauendienst, Sebastian Boring, and Heinrich Hussmann. My Phone is My Keypad: Privacy-Enhanced PIN-Entry on Public Terminals. In *Annual Conference of the Australian Computer-Human Interaction Special Interest Group*, 2009.

[DLHH10] Alexander De Luca, Katja Hertzschuch, and Heinrich Hussmann. ColorPIN: Securing PIN Entry Through Indirect Input. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.

[DLWD07] Alexander De Luca, Roman Weiss, and Heiko Drewes. Evaluation of Eye-Gaze Interaction Methods for Security Enhanced PIN-Entry. In *Australasian Conference on Computer-Human Interaction: Entertaining User Interfaces*, 2007.

[FCB10] Alain Forget, Sonia Chiasson, and Robert Biddle. Shoulder-Surfing Resistance with Eye-Gaze Entry in Cued-Recall Graphical Passwords. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.

[MVG+11] Federico Maggi, Alberto Volpatto, Simone Gasparini, Giacomo Boracchi, and Stefano Zanero. A Fast Eavesdropping Attack Against Touchscreens. In *7th International Conference on Information Assurance and Security (IAS)*, 2011.

[RRF04] Volker Roth, Kai Richter, and Rene Freidinger. A PIN-Entry Method Resilient Against Shoulder Surfing. In *ACM Conference on Computer and Communications Security (CCS)*, 2004.

[SCH70] Lionel Standing, Jerry Conezio, and Ralph N. Haber. Perception and Memory for Pictures: Single-trial Learning of 2500 Visual Stimuli. *Psychonomic Science*, 19(2):73–74, 1970.

[TA08] H. Tao and C. Adams. Pass-Go: A Proposal to Improve the Usability of Graphical Passwords. *International Journal of Network Security*, 7(2):273–292, 2008.

[TKC05] Desney S. Tan, Pedram Keyani, and Mary Czerwinski. Spy-Resistant Keyboard: More Secure Password Entry on Public Touch Screen Displays. In Ash Donaldson, editor, *OZCHI*, volume 122 of *ACM International Conference Proceeding Series*. ACM, 2005.

[TOH06] Furkan Tari, A. Ant Ozok, and Stephen H. Holden. A Comparison of Perceived and Real Shoulder-Surfing Risks Between Alphanumeric and Graphical Passwords. In *Symposium on Usable Privacy and Security (SOUPS)*, 2006.

[UDWH13] Sebastian Uellenbeck, Markus Dürmuth, Christopher Wolf, and Thorsten Holz. Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns. In *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[ZL07] Huanyu Zhao and Xiaolin Li. S3PAS: A Scalable Shoulder-Surfing Resistant Textual-Graphical Password Authentication Scheme. In *AINA Workshops (2)*, pages 467–472. IEEE Computer Society, 2007.

Phishing still works: Erfahrungen und Lehren aus der Durchführung von Phishing-Experimenten

Nadina Hintz
Friedrich-Alexander-Universität
91058 Erlangen
nadina.hintz@cs.fau.de

Markus Engelberth
Pi-One GbR
64625 Bensheim
markus@pi-one.net

Zinaida Benenson
Friedrich-Alexander-Universität
91058 Erlangen
zinaida.benenson@cs.fau.de

Felix C. Freiling
Friedrich-Alexander-Universität
91058 Erlangen
felix.freiling@cs.fau.deF

Abstract: Wir beschreiben die Durchführung und die Ergebnisse zweier Experimente, bei denen der Einfluss verschiedener Gestaltungsparameter von E-Mails und Webseiten auf den Erfolg von Phishing-Angriffen untersucht wurde. Wir berichten außerdem über unsere Erfahrungen, welche technischen, ethischen und rechtlichen Aspekte beim Design und der Durchführung solcher Experimente beachtet werden müssen.

1 Einführung

Motivation. Das Kunstwort "Phishing" bezeichnet Angriffe, die mittels betrügerischer E-Mails und manipulierter Webseiten durchgeführt werden, indem die Opfer unter einem Vorwand zur Eingabe von Passwörtern, Konto- und Kreditkarteninformationen oder anderen sensiblen Daten aufgefordert werden. Das kriminelle Ziel dieser Angriffe ist, mit den so erlangten Daten Identitätsdiebstahl zu betreiben. Angriffe auf Unternehmen haben häufig unternehmensinterne Daten oder Kundendaten zum Ziel. Phishing kann somit einen erheblichen finanziellen Schaden verursachen oder den Ruf einer Person oder eines Unternehmens nachhaltig schädigen. Laut einer Statistik des Bundeskriminalamts [Bun12] lagen die Schäden, die in Deutschland durch Phishing im Bereich Online-Banking entstanden sind, von 2010 bis 2012 zwischen 13,8 und 25,7 Mio. Euro pro Jahr. In den USA bewegt sich der jährliche Schaden nach unterschiedlichen Einschätzungen zwischen 61 Millionen und 3 Milliarden US-Dollar [Hon12].

Phishing ist ein Kriminalitätsfeld, das auf vielen Ebenen angegangen wird. So arbeiten etwa die Strafverfolgungsbehörden an der Zerschlagung der Phishing-Infrastrukturen wie z.B. Botnetzen. Andere Organisationen, wie etwa die Anti Phishing Working Group (APWG) [APWG13] in den USA haben das Ziel, die Aktivitäten von Behörden und privaten Akteuren besser zu koordinieren. Bekannt sind auch Freiwilligenorganisationen wie PhishTank [Phi13], die Blacklisten mit Webadressen pflegen, an denen sich Phishing-Seiten befinden. In Deutschland befasst sich die Arbeitsgruppe „Identitätsschutz im In-

ternet“ [ai3] mit technischen und rechtlichen Fragen zu Phishing und Identitätsdiebstahl. Es gibt auch eine Vielzahl von technischen Werkzeugen zur Erkennung von Phishing-Angriffen, und auch Internet- und E-Mail-Provider bieten mittlerweile Anti-Phishing-Dienste an. Diese Tools können jedoch bei weitem nicht alle Angriffe erkennen. So untersuchten Zhang et al. [ZECH07] 10 Anti-Phishing-Tools, wovon nur ein Tool 90 % der Phishing-Webseiten korrekt identifizierte und die meisten Tools weniger als 50% der Phishing-Webseiten erkannt haben.

Das mutmaßlich relevanteste Problem im Kontext von Phishing ist jedoch der „Faktor Mensch“. Einerseits sind Anti-Phishing-Tools oft nicht nutzerfreundlich gestaltet, so dass die Personen nicht verstehen, um welches Problem es sich handelt, wo die Gefahr liegt und warum sie z.B. nicht auf einen Link klicken sollten [Hon12]. Deswegen reagieren Opfer in vielen Fällen nicht angemessen auf Warnungen, bemerken sie nicht oder halten diese für ungültig [WMG06]. Andererseits nutzen auch die raffiniertesten technischen Gegenmaßnahmen nichts, wenn Menschen von der Authentizität einer Phishing-Mail überzeugt sind. Hong [Hon12] schreibt hierzu:

> „*It does not matter how many firewalls, encryption software, certificates or two-factor authentication mechanisms an organization has, if the person behind the keyboard falls for a phish.*“

Die Qualität der Manipulation eines Opfers ist somit ein wesentlicher Einflussfaktor auf den Erfolg der Phishing-Angriffe. Da das menschliche Verhalten von verschiedenen Faktoren, wie z.B. der aktuellen Lebenssituation, den gemachten Erfahrungen und der natürlichen Neugier abhängt, ist es besonders schwer, geeignete Schutzmechanismen zu entwickeln.

Forschungsbeitrag. Studien, die den Einflussfaktor bestimmter Aspekte auf den Erfolg von Phishing-Angriffen untersuchten, wurden bisher größtenteils in den USA im universitären Umfeld durchgeführt, die meisten Teilnehmer waren Studierende oder Universitätsmitarbeiter (die verwandten Arbeiten werden in Abschnitt 2 genauer beschrieben). In dieser Arbeit werden zwei Phishing-Experimente vorgestellt, die in Deutschland durchgeführt wurden und sich auch auf das Unternehmensumfeld erstreckten. Aus den Erfahrungen dieser beiden Studien beschreiben wir weiterhin, welche technischen, ethischen und rechtlichen Aspekte beim Design und der Durchführung solcher Experimente eine Rolle spielen, und wie diese Experimente korrekt umgesetzt werden können. Wir möchten damit auch eine Diskussion im deutschsprachigem Raum über die sinnvolle und korrekte Durchführung von Phishing-Experimenten anstoßen.

Das erste Experiment wurde in Zusammenarbeit mit vier deutschen Unternehmen durchgeführt und untersuchte den Einfluss des Absenders (extern oder unternehmensintern), des Inhalts und der Formulierung einer Phishing-Mail und des Designs der entsprechenden Webseite auf den Erfolg der Phishing-Angriffe. Nicht überraschend (aber trotzdem beunruhigend) war das Ergebnis, dass auch eine unpersönlich formulierte E-Mail, die einen Link zu einem (nicht existierenden) reißerischen Artikel enthält, eine Klickrate von mehr als 10% verursacht hat. Durch eine persönliche Ansprache und eine Aufforderung zur Änderung der unternehmensinternen Zugangsdaten wurde die Anfälligkeit für den Angriff fast verdoppelt.

Das zweite Experiment fand an einer deutschen Universität statt und untersuchte die Reaktion der Studierenden auf einen Phishing-Angriff, der absichtlich möglichst „einfach“ gestaltet wurde: keine persönliche Anrede, kein besonders ausgeklügelter Vorwand, und eine leere Webseite, die nur Eingabefelder für Login und Passwort enthält. Überraschenderweise haben auch hier mehr als 10 % der Teilnehmer versucht, ihre Daten einzugeben.

Ausblick. Dieser Artikel ist folgendermaßen organisiert. Im Abschnitt 2 werden verwandte Arbeiten aufgeführt. Im Abschnitt 3 werden die durchgeführten Studien und deren Ergebnisse vorgestellt. Im Abschnitt 4 diskutieren wir wichtige Aspekte bei der Durchführung von Phishing-Studien. Anschließend werden im Abschnitt 5 unsere Ergebnisse zusammengefasst.

2 Verwandte Arbeiten

Einflussfaktoren auf die Anfälligkeit für Phishing-Angriffe. Erste Arbeiten zu psychologischen Einflussfaktoren untersuchten, nach welchen Kriterien die Nutzer Phishing-Angriffe erkennen. Downs et al. [DHC06] analysierten mit Hilfe von Interviews und Rollenspielen, nach welchen Entscheidungskriterien Personen E-Mails als gefährlich einstufen. Dhamija et al. [DTH06] erforschten, welche Strategien die Benutzer einsetzen um zu entscheiden, ob eine Webseite echt oder gefälscht ist. Jakobsson und Ratkiewicz haben sogenannte „context-aware“ Experimente durchgeführt, indem sie öffentlich verfügbare Daten von eBay-Nutzern und die typischen Kommunikationsmuster bei eBay verwendet haben, um die Glaubwürdigkeit der Phishing-Mails zu erhöhen. Jagatic et al. [JJJM07] entnahmen Informationen über soziale Kontakte aus sozialen Netzwerken und haben anschließend eine simulierte Phishing-Mail so dargestellt, als komme sie von einem Bekannten, was den Erfolg der Phishing-Angriffe drastisch verbesserte im Vergleich zu unpersönlich formulierten E-Mails.

Ergebnisse dieser Arbeiten haben gezeigt, dass es den Nutzern schwer fällt, Phishing zu erkennen und dass ihre Entscheidungskriterien nicht angemessen sind. Als Blythe et al. [BPC11] fünf Jahre später ähnliche Untersuchungen durchführten, haben sie herausgefunden, dass sich nicht viel geändert hat. Nach wie vor achten die Benutzer bei E-Mails auf den Absender („wenn ich bei dieser Firma Kunde bin, oder wenn ich den Absender persönlich kenne, dann ist die E-Mail echt“), auf das Design und die Sprache der E-Mails und Webseiten („wenn keine offensichtlichen Sprachfehler auffallen, vertraute Markenzeichen präsent sind und alle Links funktionieren, dann ist die E-Mail oder die Webseite echt“) und haben Schwierigkeiten, technische Details zu interpretieren, wie z.B. den Aufbau der Links oder die Präsenz von „padlock icons“.

Der Einfluss der demographischen Unterschiede (Alter und Geschlecht) auf den Umgang mit Phishing wurde in mehreren Arbeiten gemessen, meistens ebenfalls im universitären Umfeld. Während einige Untersuchungen keine Korrelation zwischen demographischen Faktoren und der Anfälligkeit für Phishing finden konnten [DTH06, SMK+07], haben andere Experimente signifikante Unterschiede festgestellt [JJJM07, SHK+10, BPC11]. So

waren jüngere Personen (18-25 Jahre) anfälliger für Phishing-Attacken, und Frauen waren anfälliger als Männer. Der Einfluss des demographischen Merkmals „Bildung“ wurde dagegen noch nicht ausreichend untersucht.

Die Wirksamkeit von Anti-Phishing Tools und Warnungen wurde in einer Reihe von Arbeiten untersucht. Dhamija et al.[DTH06] fanden heraus, dass die meisten Benutzer die passiven Browser-Hinweise auf die Vertrauenswürdigkeit einer Webseite („security indicators“) nicht beachten und daher viele Phishing-Webseiten als echt einstufen. Egelman et al. [ECH08] haben untersucht, ob aktive und passive Warnungen beim Öffnen einer Phishing-Webseite das Verhalten der Nutzer beeinflusst. 79% der Nutzer haben aktive Warnungen und 13% der Nutzer haben passive Warnungen beachtet. Lin et al. [LGT+11], Kirlappos et al. [KS12] und Li et al. [LHB12] haben die Benutzbarkeit von weiteren Anti-Phishing Maßnahmen getestet. Die Ergebnisse dieser Studien haben gezeigt, dass nur aktive Warnungen zu einer signifikanten Reduktion der Anzahl der Phishing-Opfer führen, wobei die Verständlichkeit der Warnungen und der Indikatoren nicht ausreicht, um die Nutzer bei ihren Entscheidungen ausreichend zu unterstützen.

Benutzerschulungen sind eine weitere Maßnahme zum Schutz gegen Phishing. Die an der CyLab der CMU entwickelten „Anti-Phishing Phil“ [SMK+07] und „PhishGuru“ [KCA+09] sind die am meisten bekannten Systeme zur Unterstützung von Nutzer-Schulungen. In einer vergleichender Untersuchung der beiden Systeme [KSA+10] fanden die Entwickler heraus, dass die beiden Schulungsmaßnahmen die Anzahl der Phishing-Opfer reduziert haben. Allerdings zeigten die Nutzer keine intrinsische Motivation auf, etwas über den sicheren Umgang mit Phishing-Angriffen zu lernen. Wahrscheinlich hängt die Wirksamkeit der Schulungen hochgradig vom Umfeld ab, in dem diese durchgeführt werden.

Ethische Aspekte. Neben allgemeinen Diskussionen zur ethischen Orientierung im Bereich der IT-Sicherheitsforschung [DBD11] gibt es auch Literatur, die sich spezifisch mit der Durchführung von Phishing-Experimenten beschäftigt hat. So beschreiben Jakobsson et al. [JJF08] drei Möglichkeiten, um Phishing-Experimente durchzuführen: (1) mit Hilfe von Umfragen, (2) durch Labor-Experimente oder (3) durch lebensnahe Experimente. Gleichzeitig führen die Autoren auch die Vor- und Nachteile dieser Möglichkeiten auf. Die Autoren argumentieren, dass lebensnahe Experimente trotz der ethischen Nachteile die beste Möglichkeit bieten, das reale Verhalten von Personen zu messen. Nur so könnten neue Erkenntnissen erzielt werden, welche Entwicklern helfen, die Maßnahmen gegen Phishing zu verbessern. Als Beispiel für ethisch korrekt durchgeführte Experimente nennen sie ihre eigenen Arbeiten [JR06, JJJM07].

Schrittweiser et al. [SMW13] stellen diese Art von Studiendesign in Frage und kritisieren das ethische Vorgehen einiger Forscher, u.a. Jagatic et al. [JJJM07], bei der Durchführung von Phishing-Studien. Die Autoren führen auf, dass die Forscher gewährleisten müssen, dass die Probanden durch die Studiendurchführung keinen Schaden nehmen und dass die Forscher sich bewusst sein sollten, dass ihre Forschungsergebnisse von Kriminellen missbraucht werden könnten. Sie diskutieren grundlegende ethische Prinzipien der Forschung und schlagen vor, dass die wissenschaftliche Gemeinschaft der IT-Sicherheitsforscher ethisch verpflichtende Standards entwickelt, vergleichbar mit denen in der Medizin.

3 Durchgeführte Studien

Wir stellen zwei Phishing-Studien vor. Die erste Studie wurde in vier deutschen Unternehmen durchgeführt mit dem Ziel, den Grad der Sensibilisierung der Mitarbeiter zu messen. Die zweite Studie fand an einer deutschen Universität statt. In beiden Studien wurden einige Faktoren gemessen, die die Anfälligkeit für Phishing beeinflussen. Nachdem der Aufbau und die Ergebnisse der Studien in diesem Abschnitt vorgestellt werden, diskutieren wir im Abschnitt 4 wichtige Aspekte der Durchführung der Phishing-Studien anhand unserer Erfahrungen.

3.1 Studie 1: Phishing-Experimente im Unternehmensumfeld

3.1.1 Aufbau der Studie 1

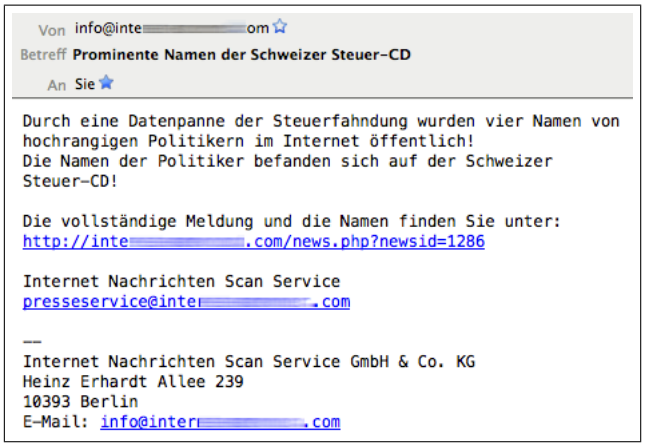
In Zusammenarbeit mit einem Unternehmen aus der IT-Sicherheitsbranche haben wir über einen Zeitraum von einem Jahr eine experimentelle Studie durchgeführt mit dem Ziel, die Reaktionen der Mitarbeiter deutscher Unternehmen auf den Erhalt von Spam- und Phishing-Mails zu messen. Nach umfangreicher Teilnehmerakquise haben sich vier Unternehmen, die zum Zeitpunkt der Studie zwischen 103 und 425 Mitarbeiter beschäftigten, dazu bereit erklärt, an dieser Studie teilzunehmen.

Die Motivation der teilnehmenden Unternehmen bestand darin, quantitative Informationen über das Sicherheitsbewusstsein ihrer Mitarbeiter zu erhalten. Nach Abschluss der Studie haben wir den teilnehmenden Unternehmen jeweils einen Abschlussbericht vorgelegt. Alle Unternehmen haben die Abschlussberichte genutzt, um ihren Mitarbeitern in Form von Vorträgen die Ergebnisse zu präsentieren. Vor Durchführung der Studie haben wir mit allen beteiligten Unternehmen die geplanten Spam- und Phishing-Mails abgesprochen und die jeweiligen Systemadministratoren über die Durchführung der Studie informiert.

Im Rahmen der Experimente wurden zwei Arten von E-Mails versandt. Die erste E-Mail war eine klassische Spam-Mail mit einem Verweis auf eine externe Internetseite. Anhand der Spam-Mail lässt sich die Rate der Empfänger ermitteln, die bereits dem Verweis einer unpersönlichen, reinen Spam-Mail folgen. Dies ist deshalb ein hochinteressanter Wert, da das Anklicken eines Links das initiale und notwendige Ereignis eines erfolgreichen Phishing-Betrugs darstellt. Die zweite E-Mail war personalisiert, d. h. die Empfänger wurden namentlich angesprochen. Zudem war sie stark auf das jeweilige Unternehmen zugeschnitten, so dass sie das andere Ende des Spektrums möglicher Phishing-Angriffe darstellt: so genanntes *Spear-Phishing*. Nachfolgend werden die beiden verwendeten E-Mail-Arten detailliert beschrieben.

E-Mail 1: Spam mit einem Verweis auf eine Internetseite. Die E-Mails enthielten einen Verweis, der auf eine von uns erstellte Webseite führte. Die E-Mails waren nicht personalisiert, d. h. die Empfänger wurden nicht namentlich angesprochen. Der Inhalt der E-Mails war jeweils an ein aktuelles Ereignis angelehnt, das zum Zeitpunkt des E-Mail-

Versands in den Medien diskutiert wurde. Somit sollte ein möglichst großer Empfängerkreis angesprochen werden. Ein Beispiel einer solchen E-Mail befindet sich in Abbildung 1. In unserer Studie führte der Verweis zu einer harmlosen Webseite, die den HTTP-Fehler 504 (Gateway Timeout) nachahmte. Die Verweise in den E-Mails enthielten alle eine anonyme Benutzer-ID (in dem Beispiel der Abbildung als News-ID getarnt), so dass wir die Klicks einzelner Empfänger unterscheiden konnten. Als Absender haben wir eine erfundene Internet-Nachrichtenagentur verwendet.


Von info@inte[redacted]om
Betreff **Prominente Namen der Schweizer Steuer-CD**
An Sie

Durch eine Datenpanne der Steuerfahndung wurden vier Namen von hochrangigen Politikern im Internet öffentlich!
Die Namen der Politiker befanden sich auf der Schweizer Steuer-CD!

Die vollständige Meldung und die Namen finden Sie unter:
http://inte[redacted].com/news.php?newsid=1286

Internet Nachrichten Scan Service
presseservice@inter[redacted].com

--
Internet Nachrichten Scan Service GmbH & Co. KG
Heinz Erhardt Allee 239
10393 Berlin
E-Mail: info@inter[redacted].com

Abbildung 1: Beispiel einer klassischen Spam-Mail, die im Rahmen der Studie 1 verschickt wurde.

Mail 2: Phishing-Mail zur Änderung eines internen Passworts. In den von uns versandten Phishing-Mails wurden die Empfänger dazu aufgefordert, aus Sicherheitsgründen ihre firmeninternen Zugangsdaten zu ändern. Um die Glaubwürdigkeit dieser Aufforderungen zu erhöhen, wurde als Absenderadresse die E-Mail-Adresse des Mitarbeiters des Unternehmens verwendet, der für solche administrativen Aufgaben zuständig ist. Die Empfänger der Nachrichten wurden mit Nachnamen angesprochen. In den Phishing-Mails war ein Verweis enthalten, der zu einer von uns erstellten Webseite führte, auf der die Mitarbeiter ihr altes und ein neues Passwort eingeben konnten. Solch eine Seite ist exemplarisch in Abbildung 2 dargestellt. Dabei wurde eine unternehmensfremde Domain verwendet. Um die Glaubwürdigkeit der Verweise zu erhöhen, haben wir der Domain Subdomains vorangestellt, die an den Namen des entsprechenden Kunden angelehnt waren – um dadurch wiederum die Glaubwürdigkeit der gesamten E-Mail zu erhöhen.

Die von uns erstellten Internetseiten wurden allesamt auf Servern des IT-Sicherheits-Unternehmens gehostet, mit dem wir in Kooperation die Studie durchgeführt haben. Von den Eingaben auf den „Phishing“-Seiten haben wir neben den Benutzernamen, Zeitstempeln und Benutzer-IDs auch gesalzene Hashwerte der eingegebenen Passwörter gespeichert. Auf diese Weise konnten wir Kennwörter voneinander unterscheiden ohne diese im Klartext zu kennen. Durch das Salz wird zudem verhindert, dass man durch einen Brute-Force-Angriff an zu kurze Passwörter im Klartext kommt. Auch die Kommunikation zwischen

Abbildung 2: Beispiel einer von uns erstellten Phishing-Seite, die im Rahmen der Studie 1 für einen der vier Kunden verwendet wurde.

Phishing-Seite und Webbrowser war durch die Verwendung eines Server-Zertifikats verschlüsselt, so dass auch hier die Passwörter nicht im Klartext übertragen wurden.

Alle von uns aufgezeichneten Daten haben wir nach der Übergabe der anonymisierten Auswertung der Ergebnisse wieder gelöscht. Die teilnehmenden Unternehmen hatten keinen Zugriff auf Rohdaten.

3.1.2 Ergebnisse der Studie 1

Die von uns durchgeführten Experimente haben gezeigt, dass wir „bessere“ Ergebnisse erzielen konnten, je glaubwürdiger wir die Phishing- und Spam-Mails gestaltet haben. Das Spoofen der Absenderadresse und das Erstellen einer „Phishing“-Seite, die optisch fast identisch zur Originalseite ist, haben einen erheblichen Einfluss auf die erzielten Ergebnisse. Während bei der ersten E-Mail nur ca. 12 % der Empfänger dem Verweis innerhalb der Spam-Mail gefolgt sind, hätten wir bei der zweiten E-Mailart von fast 20 % der Empfänger eingegebene Passwörter aufzeichnen können. Jeweils innerhalb der ersten 2 Stunden nach Mail-Versand konnten die meisten Klicks registriert werden, und nach durchschnittlich 5,83 Tagen wurden 90 % aller Reaktionen aufgezeichnet. Tabelle 1 fasst die Ergebnisse der Studie zusammen.

3.2 Studie 2: Phishing unter Universitätsstudenten

Eine zweite Studie wurde an einer deutschen Universität durchgeführt mit dem Ziel herauszufinden, ob die Teilnehmer einem Link folgen, der auf eine offensichtlich gefälschte Webseite führt und dort ihre Zugangsdaten eingeben. Bei Studie 2 war die „Phishing“-Webseite nicht aufwendig gestaltet. Es war lediglich eine leere Webseite mit zwei Eingabefeldern für den Nutzernamen und das Passwort. Die Empfänger wurden bei dieser E-Mail nicht mit Namen angesprochen.

		Mail 1	**Mail 2**
Unternehmen	Anzahl der Mitarbeiter	Dem Verweis der Spam-Mail gefolgt	Eingaben auf „Phishing"-Webseite gemacht
1	186	15 *(8,06 %)*	24 *(12,90 %)*
2	425	65 *(15,29 %)*	84 *(19,76 %)*
3	112	7 *(6,25 %)*	28 *(25,00 %)*
4	103	11 *(10,68 %)*	25 *(24,27 %)*
Gesamt	826	98 *(11,86 %)*	161 *(19,49 %)*

Tabelle 1: Übersicht über die Ergebnisse der Experimente in Unternehmen. Hinter den absoluten Zahlen ist jeweils der Prozentanteil der Mitarbeiter des jeweiligen Unternehmens angegeben, von dem wir eine Reaktion aufzeichnen konnten.

Als Absender der E-Mail fungierte der Paketlieferservice Hermes mit der Domain *no-reply@hermes-support.de* (die eigentliche Domain von Hermes heißt *myhermes.de*). Den Empfängern wurde mitgeteilt, dass ein Paket nicht zugestellt werden konnte und dass sie sich für eine erneute Zulieferung unter dem beigefügten Link auf der Webseite anmelden müssen. Jedem Link wurde wie bei Studie 1 ein individualisierter Hashwert beigefügt, so dass wir Klicks unterschiedlicher Empfänger unterscheiden konnten. Wir haben gespeichert, wie oft die Probanden auf den Link geklickt haben und, falls sie ihre Zugangsdaten eingegeben haben, wie oft sie dies wiederholt getan haben. Falls die Probanden versucht haben, ihre Daten einzugeben, haben sie eine Fehlermeldung angezeigt bekommen.

Für die Studie wurden 919 Studenten über einen E-Mail-Verteiler rekrutiert. Um die Studenten in ihrem Verhalten nicht zu beeinflussen, haben wir ihnen zunächst mitgeteilt, die Studie behandle das Thema „Nutzerverhalten im Internet". Insgesamt sind 50 % aller Studienteilnehmer dem Link in der Hermes E-Mail gefolgt und 11 % haben ihre Nutzerinformationen eingegeben. 29 % aller Empfänger haben mehrfach auf den Link geklickt und einige haben ihre Nutzerdaten erneut angegeben, obwohl die erste Eingabe nicht funktionierte. Genau wie bei Studie 1 konnten die meisten Klicks innerhalb der ersten 2 Stunden gemessen werden. Tabelle 2 fasst die Ergebnisse der Studie zusammen.

Teilnehmer	Anzahl der Teilnehmer	Dem Verweis auf Hermes gefolgt	Nutzerdaten eingegeben	Mehrfach auf Link geklickt
Frauen	596	375 *(62,92 %)*	69 *(11,58 %)*	223 *(37,42 %)*
Männer	323	85 *(26,32 %)*	34 *(10,53 %)*	46 *(14,24 %)*
Gesamt	919	460 *(50,05 %)*	103 *(11,21 %)*	269 *(29,27 %)*

Tabelle 2: Übersicht über die Ergebnisse des Experiments an der Universität. Hinter den absoluten Zahlen ist jeweils der Prozentanteil der Teilnehmer angegeben, von dem wir eine Reaktion aufzeichnen konnten.

3.3 Zusammenfassung der Ergebnisse

Die Ergebnisse beider Studien belegen erneut: *Phishing still works.* Insgesamt haben die Angestellten von Unternehmen und die Studierenden ähnliche Anfälligkeitsrate und Reaktionszeit aufgewiesen. Auf „einfachere“ Phishing-Versuche haben ca. 10 % der Empfänger reagiert, und der Großteil der Reaktionen konnte innerhalb der ersten beiden Stunden gemessen werden. Die meisten im Abschnitt 2 angeführten Arbeiten kamen zu ähnlichen Ergebnissen.

Dies bedeutet für den Angreifer, falls seine Webseite nach einiger Zeit geblacklistet wird, dass er in den ersten zwei Stunden bereits eine große Erfolgschance hat, denn laut den Angaben des Phishing-Blacklistenbetreibers PhishTank [Phi13] brauchte dieser im Jahr 2012 im Mittel 2 Stunden und 54 Minuten, um eine Webseite als bösartig zu identifizieren.

4 Lehren aus der Durchführung unserer Experimente

Wir sind bei der Durchführung unserer Experimente auf vielfältige Probleme gestoßen. Einige davon waren erwartet worden, andere waren unerwartet. In diesem Abschnitt beschreiben wir einige Lehren, die wir dokumentieren und weitergeben wollen.

4.1 Nutzen der Phishing-Experimente

Insbesondere ökonomische Anreize können hier eine Rolle spielen: Die Aufwendungen (sowohl monetärer als auch zeitlicher und organisatorischer Art), die aufgrund der Durchführung der Experimente entstehen, fallen in der Regel wesentlich geringer aus als Folgeschäden, die ein Phishing-Betrug innerhalb der Belegschaft nach sich ziehen würde.

Spätestens wenn in einer Organisation nach einem solchen Experiment das Fehlen eines klar definierten Vorgehens bei Phishing-Angriffen bewusst wird (wie kann man diese Angriffe erkennen, was tun beim Verdacht auf einen Angriff, wer und wie warnt die Mitarbeiter), wird der Nutzen dieser Experimente klar erkennbar.

4.2 Kommunikation mit beteiligten Institutionen

Es gibt viele verschiedene Institutionen, die von der Durchführung einer Phishing-Studie betroffen sind. Zuvorderst ist natürlich die Organisation zu nennen, deren Mitglieder die Zielgruppe des Experiments sind. Meist ist dies der Auftraggeber der Studie, aber auch unabhängig davon ist es notwendig, dass diese Organisation mit der Durchführung einverstanden ist. Eine andere betroffene Institution ist diejenige, in deren Namen die Phishing-Mails versendet werden. Dies kann der Auftraggeber sein oder nicht. Schließlich ist diejenige Institution noch betroffen, über die die Phishing-Mails verschickt werden bzw. die die

Phishing-Seite auf ihren Servern vorhält. Dies kann ebenfalls der Auftraggeber sein aber auch eine davon unabhängige Organisation (z.B. ein anderer E-Mail-Provider). Inwiefern diese Institutionen in die Kommunikation eingebunden werden können oder sollten, wird weiter unten diskutiert.

Mit den entsprechenden Institutionen sollten die notwendigen Fragen zum Schutz personenbezogener Daten geklärt werden, also welche Daten für welchen Zeitraum erhoben und gespeichert werden. Dies sollte Teil einer ausführlichen Studiendokumentation sein, die mit der betroffenen Institution abgestimmt werden muss. Aus der Beschreibung der Studie sollte hervorgehen, welche Daten gespeichert werden, ob personenbezogene Daten gespeichert werden, wo diese Daten gespeichert und wann sie wieder gelöscht werden, von welchem Netzwerk aus und unter welchem Absender E-Mails versandt werden und was der Inhalt der Nachrichten sein wird.

Unserer Erfahrung nach kommt es im Rahmen von Phishing-Experimenten fast immer zu Situationen, in denen Mitarbeiter vermehrt eine zuständige Person oder Abteilung auf die Geschehnisse aufmerksam machen. Deshalb sollten also insbesondere alle Institutionen darüber informiert werden, wann die Nachrichten versandt werden, damit ggf. eine erhöhte Anzahl an Beschwerden oder Warnhinweisen der Studienteilnehmer entgegengenommen werden kann. Ebenfalls haben wir erlebt, dass ein aufmerksamer Abteilungsleiter etliche Mitglieder seiner Abteilung vor dem Öffnen unserer E-Mail gewarnt hat.

Bei Studien an Universitäten sollten das Rechenzentrum, der/die Datenschutzbeauftragte, die Ethikkommission (falls vorhanden) und das HelpDesk über die Durchführung informiert sein und dieser zustimmen. Bei Studien an Unternehmen sollten der Geschäftsführer, der Betriebsrat, die IT-Revision und/oder die zuständigen Systemadministratoren über die Durchführung informiert sein und dieser zustimmen.

Schließlich sind noch die eigentlichen Teilnehmer der Studie in die Betrachtungen mit einzubeziehen. Eine explizite Information dieser Personen über den Sinn und Zweck der durchgeführten Experimente ist natürlich nicht sinnvoll, da sie dem Zweck der Studie widerspricht. Trotzdem sollte in irgend einer Form eine Art Einverständnis dieser Personengruppe herbeigeführt werden, präferiert entweder indirekt über den Vorgesetzten (die beauftragende Organisation) oder durch eine allgemeine Zustimmung zu einem “Experiment zur Internetnutzung”. Wir sind uns bewusst, das letzteres ethische Schwierigkeiten mit sich bringt. Dies sind jedoch dieselben Schwierigkeiten, mit denen auch viele Experimente im Bereich der Psychologie behaftet sind.

Alle Studienteilnehmer sollten jedoch *nach* Durchführung der Studie über diese vollständig aufgeklärt werden. Allen betroffenen Institutionen sollte ein Abschlussbericht mit den Ergebnissen vorgelegt werden. Der Zeitpunkt der Aufklärung sollte bereits vor Durchführung der Studie feststehen. In welcher Form die Studienteilnehmer über die Durchführung der Studie informiert werden, sollte in jeder Institution individuell festgelegt werden (z.B. per E-Mail oder in einem Vortrag). Aus der Aufklärung sollten das Ziel und der Zweck sowie ein eindeutiger Ansprechpartner hervorgehen, damit Studienteilnehmer die Möglichkeit haben, weitere Fragen zu klären oder Feedback zu geben.

4.3 Technische Aspekte

Phishing-Studien sind nicht nur organisatorisch, sondern auch technisch eine Herausforderung. Viele Institutionen wie Universitäten und Unternehmen haben technische Maßnahmen (Spamfilter, automatische Entfernung von Anhängen), um den Eingang von Phishing-Mails zu verhindern. Diese technischen Maßnahmen sollten im Zuge dieser Studie nicht abgeschaltet werden. Es muss jedoch auch gewährleistet sein, dass die Phishing-Mails, die Teil der Studie sind, auch zuverlässig ihre Adressaten erreichen. Vor der Umsetzung einer Studie sollten die Phishing-Mails also an unabhängigen Personen getestet werden, damit eventuell auftretende Probleme frühzeitig erkannt werden.

Eine besondere technische Herausforderung stellt der Umgang mit Passwörtern dar, die von den Teilnehmern eventuell eingegeben werden. Entweder sollen diese überhaupt weder übertragen noch gespeichert werden oder, falls der Studienzweck dies erfordert, unbedingt verschlüsselt übertragen, als gesalzener Hashwert gespeichert und so schnell wie möglich gelöscht werden. Der Zeitpunkt der Löschung soll schon vor Beginn der Studie feststehen.

Eine andere Möglichkeit zum Umgang mit Passwörtern besteht, wenn die Experiment-Teilnehmer, die auf den Link in der simulierten Phishing-Mail klicken, zur echten Webseite umgeleitet werden und dort ihre Daten eingeben [JR06, JJJM07]. Das ist allerdings nur dann möglich, wenn die Eingabe der Login-Daten auf der echten Webseite registriert und von „normalen“ Login-Vorgängen unterschieden werden kann.

4.4 Rechtliche Aspekte

Die rechtliche Würdigung von Phishing-Experimenten kann in einem Beitrag wie diesem nicht abschließend behandelt werden. Wir können hier nur mögliche Problemfelder anreißen und Argumente sammeln, die in zukünftigen Diskussionen noch vertieft werden müssen.

Aus strafrechtlicher Sicht kann man beim Versand von Phishing-Mails die Täuschung durch den gefälschten Absender unter § 269 Abs. 1 StGB (Fälschung beweiserheblicher Daten) fassen. Weidemann schreibt dazu:

> Beim Versenden von “Phishing-E-Mails” ist dies ebenfalls der Fall, weil die E-Mail nach dem Absenden jedenfalls beim Empfänger nach dem Aufrufen gespeichert und dem Empfänger vorgetäuscht wird, es handele sich bspw. um eine E-Mail eines Vertragspartners. [Wei10b, Rdn. 9 m.w.N.]

Allerdings schränkt die Norm den Tatbestand derart ein, dass er „zur Täuschung im Rechtsverkehr“ ausgeübt werden muss. Rechtlich bedeutet das, dass er „auf die Herbeiführung eines Irrtums bei dem Getäuschten sowie die Veranlassung des Getäuschten zu einem rechtserheblichen Handeln“ [Wei10a, Rdn. 29] gerichtet sein muss. [Erb03, Rdn. 205] führt dazu aus:

> Nicht rechtserheblich ist ein Sachverhalt dann, wenn im Einzelfall (selbst bei Aufdeckung der Täuschung) sicher auszuschließen ist, dass der Getäuschte ein Verhalten an den Tag legen wird, das über eine Reaktion im zwischenmenschlichen Bereich hinaus die Erfüllung einer (vermeintlichen) Rechtspflicht oder die Einforderung eines (vermeintlichen) Rechts beinhaltet.

Da es nicht unsere Absicht ist, dass die Probanden rechtserheblich handeln (und wir dies auch durch geeignete technische Maßnahmen ausschließen), erscheint § 269 StGB nicht einschlägig. Andere strafrechtliche Vorschriften der Computerkriminalität, etwa §§ 202a–202c oder 303a–303b, sind auf diesen Sachverhalt ebenfalls nicht anwendbar, es sei denn, Passwörter werden tatsächlich im Klartext ausgespäht und abgespeichert. Dies muss durch technische Vorkehrungen wie eine verschlüsselte Verbindung für die Kommunikation sowie Speicherung gesalzener Passwort-Hashes soweit wie irgend möglich ausgeschlossen werden.

Der Bereich des Zivilrechts erscheint hier ergiebiger. Beispielsweise könnte die Organisation, in dessen Namen E-Mails verschickt werden, die Verletzung des Namensrechts (§ 12 BGB) geltend machen. Auch müssen die AGBs oder Benutzerrichtlinien der Organisation, über die die E-Mails versendet werden und bei der die Phishing-Seite vorgehalten wird, beachtet werden. Auf die Datenschutzrisiken sind wir weiter oben bereits eingegangen. Ohne die Zustimmung der Betroffenen könnte der Versand von Phishing-Mails in all diesen Fällen zu Abmahnungen und/oder Unterlassungserklärungen führen. Da man in der Praxis von Firmen wie Facebook oder Hermes wohl kaum eine offizielle Zustimmung erhalten wird, muss dieses Risiko in Betracht gezogen und durch einen begrenzten Adressatenkreis, dem Hinweis auf die Umstände der Forschung usw. minimiert werden.

Trotz all dieser Vorkehrungen kann es passieren, dass die Studienteilnehmer durch die Studie zu Schaden kommen. Einem Studienteilnehmer waren beispielsweise durch eine Phishing-Mail dadurch Kosten entstanden, dass er einen Computerspezialisten aufgesucht hatte aus Angst, sein Computer sei mit Schadsoftware infiziert. Dem kann beispielsweise durch unmittelbare Aufklärung im Anschluss an das Experiment vorgebeugt werden.

4.5 Ethische Aspekte

Die Durchführung solcher Experimente trifft insbesondere im Unternehmensumfeld auf starke ethische Vorbehalte, wobei diese Vorbehalte auch für die Durchführung der Phishing-Experimente in anderen Umgebungen gelten. Bedenken bestehen beispielsweise dagegen, dass einzelne Mitarbeiter durch die Experimente bloßgestellt werden könnten.

Bedenklich könnte auch die Tatsache sein, dass sich Mitarbeiter einer Firma überwacht fühlen könnten, wenn die Firmenleitung entsprechenden Experimenten zustimmt. Diese Bedenken können wir natürlich nicht komplett ausräumen, denn es ist schwer vorauszusagen, was die Mitarbeiter fühlen, wenn sie über die Studie aufgeklärt werden. Durch die anonymen Benutzer-IDs war jedoch bereits sichergestellt, dass kein einzelner Mitarbeiter denunziert werden kann. Zudem haben wir mit den Unternehmen abgesprochen, dass sie ihre Mitarbeiter nach der Studie über diese selbst aufklären, um Transparenz zu schaffen.

Einerseits kann man zwar leider nicht ausschließen, dass sich die Mitarbeiter trotzdem „ertappt“ gefühlt haben. Andererseits haben die Unternehmen mit der Durchführung der Studie eine gute Möglichkeit, die Mitarbeiter zu sensibilisieren, denn der Schulungseffekt einer solchen Maßnahme kann als besser eingestuft werden als der Effekt einer theoretischen Belehrung. Außerdem könnte man argumentieren, dass auch für einen Mitarbeiter die Folgen eines Fehlers bei einem simulierten Angriff viel harmloser sind, und die Mitarbeiter durch die nach dem Experiment folgende Aufklärung vor Anfälligkeit für echte Angriffe geschützt werden.

Wichtig ist auf jeden Fall, dass durch die Phishing-Experimente und andere Betrugsexperimente keine Atmosphäre des allgemeinen Misstrauens in der Organisation entsteht. Dieser Punkt, genauso wie die Auswirkungen der Aufklärung über die Experimente auf einzelne Teilnehmer, wurde bisher nicht untersucht und bedarf nach unserer Meinung einer erhöhten Aufmerksamkeit.

5 Fazit und Ausblick

Unsere Ergebnisse haben gezeigt, dass nach wie vor eine viel zu hohe Anzahl an Personen Opfer von Phishing-Angriffen werden. Über die Hintergründe dieses Phänomens, d.h. welche Einflussfaktoren dazu führen, dass Personen einer Phishing-Mail oder einer Phishing-Webseite vertrauen schenken, sind sich die Forscher derzeit noch uneinig, wie in dem Kapitel 2 „Verwandte Arbeiten“ dargestellt. Aus diesem Grund ist es wichtig, auf diesem Gebiet weiter zu forschen um Wege zu finden, die Anzahl der Opfer und der Angriffe nachhaltig zu reduzieren.

Da es wahrscheinlich auch zukünftig Untersuchungen auf diesem Gebiet geben wird, ist es umso wichtiger, von den Erfahrungen der vergangenen Studien profitieren zu können. Aus diesem Grund haben wir in dieser Arbeit von unseren Erfahrungen berichtet und verschiedene Aspekte erläutert, die bei der Umsetzung solcher Studien berücksichtigt werden sollten. Wir hoffen, dass wir mit dieser Arbeit einen ersten Schritt zur Erstellung eines Grundgerüstes solcher Studien gemacht haben und unsere Forschungsergebnisse und Erfahrungen durch zukünftige Forscher ergänzt werden.

Danksagungen

Wir danken Dominik Brodowski für hilfreiche Diskussion zu den rechtlichen Aspekten von Phishing-Experimenten.

Diese Arbeit wurde unterstützt durch das Bundesministerium für Bildung und Forschung im Rahmen des Forschungsverbunds open C3S (www.open-c3s.de) und durch das Bayerische Staatsministerium für Bildung und Kultus, Wissenschaft und Kunst im Rahmen des Forschungsverbunds ForSEC (www.bayforsec.de).

Literatur

[ai3] Arbeitsgruppe Identitsschutz im Internet e.V. (a-i3). Available online at https://www.a-i3.org/; visited on November 25th 2013.

[APWG13] Inc. Anti-Phishing Working Group. Anti Phishing Working Group (APWG). Website, 2013. Available online at http://www.antiphishing.org; visited on November 25th 2013.

[BPC11] Mark Blythe, Helen Petrie und John A. Clark. F for Fake: Four Studies on How We Fall for Phish. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, Seiten 3469–3478, 2011.

[Bun12] Bundeskriminalamt. Bundeslagebild Cybercrime 2012. *Lagebilder Cybercrime*, Seite 7, 2012.

[DBD11] David Dittrich, Michael Bailey und Sven Dietrich. Building an Active Computer Security Ethics Community. *IEEE Security & Privacy*, 9(4):32–40, 2011.

[DHC06] Julie S. Downs, Mandy B. Holbrook und Lorrie Faith Cranor. Decision Strategies and Susceptibility to Phishing. In *Proceedings of the Second Symposium on Usable Privacy and Security*, SOUPS '06, Seiten 79–90, New York, NY, USA, 2006. ACM.

[DTH06] Rachna Dhamija, J. D. Tygar und Marti Hearst. Why Phishing Works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, Seiten 581–590, New York, NY, USA, 2006. ACM.

[ECH08] Serge Egelman, Lorrie Faith Cranor und Jason Hong. You'Ve Been Warned: An Empirical Study of the Effectiveness of Web Browser Phishing Warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, Seiten 1065–1074, 2008.

[Erb03] Volker Erb. *Münchener Kommentar zum StGB*, Kapitel §269. Beck, 2003.

[Hon12] Jason Hong. The State of Phishing Attacks. *Commun. ACM*, 55(1):74–81, Januar 2012.

[JJF08] Markus Jakobsson, Nathaniel Johnson und Peter Finn. Why and How to Perform Fraud Experiments. *IEEE Security and Privacy*, 6(2):66–68, Marz 2008.

[JJJM07] Tom N. Jagatic, Nathaniel A. Johnson, Markus Jakobsson und Filippo Menczer. Social Phishing. *Commun. ACM*, 50(10):94–100, Oktober 2007.

[JR06] Markus Jakobsson und Jacob Ratkiewicz. Designing Ethical Phishing Experiments: A Study of (ROT13) rOnl Query Features. In *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, Seiten 513–522, New York, NY, USA, 2006. ACM.

[KCA+09] Ponnurangam Kumaraguru, Justin Cranshaw, Alessandro Acquisti, Lorrie Cranor, Jason Hong, Mary Ann Blair und Theodore Pham. School of Phish: A Real-world Evaluation of Anti-phishing Training. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, Seiten 3:1–3:12, New York, NY, USA, 2009. ACM.

[KS12] Iacovos Kirlappos und Martina Angela Sasse. Security Education against Phishing: A Modest Proposal for a Major Rethink. *Security & Privacy, IEEE*, 10(2):24–32, 2012.

[KSA+10] Ponnurangam Kumaraguru, Steve Sheng, Alessandro Acquisti, Lorrie Faith Cranor und Jason Hong. Teaching Johnny Not to Fall for Phish. *ACM Trans. Internet Technol.*, 10(2):7:1–7:31, Juni 2010.

[LGT+11] Eric Lin, Saul Greenberg, Eileah Trotter, David Ma und John Aycock. Does Domain Highlighting Help People Identify Phishing Sites? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, Seiten 2075–2084, 2011.

[LHB12] Linfeng Li, Marko Helenius und Eleni Berki. A Usability Test of Whitelist and Blacklist-based Anti-phishing Application. In *Proceeding of the 16th International Academic MindTrek Conference*, MindTrek '12, Seiten 195–202, 2012.

[Phi13] PhisTank. PhishTank: Out of the Net, into the Tank! Website, 2013. Available online at https://www.phishtank.com; visited on November 25th 2013.

[SHK+10] Steve Sheng, Mandy Holbrook, Ponnurangam Kumaraguru, Lorrie Faith Cranor und Julie Downs. Who Falls for Phish?: A Demographic Analysis of Phishing Susceptibility and Effectiveness of Interventions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, Seiten 373–382, New York, NY, USA, 2010. ACM.

[SMK+07] Steve Sheng, Bryant Magnien, Ponnurangam Kumaraguru, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong und Elizabeth Nunge. Anti-Phishing Phil: The Design and Evaluation of a Game That Teaches People Not to Fall for Phish. In *Proceedings of the 3rd Symposium on Usable Privacy and Security*, SOUPS '07, Seiten 88–99, New York, NY, USA, 2007. ACM.

[SMW13] Sebastian Schrittweiser, Martin Mulanzzani und Edgar Weippl. Ethics in Security Research Which Lines Should Not Be Crossed? Cyber-security Ethics Dialog & Strategy Workshop (CREDS 2013), 2013.

[Wei10a] Weidemann. §267. Beck'scher Online Kommentar zum StGB, 2010.

[Wei10b] Weidemann. §269. Beck'scher Online Kommentar zum StGB, 2010.

[WMG06] Min Wu, Robert C. Miller und Simson L. Garfinkel. Do Security Toolbars Actually Prevent Phishing Attacks? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, Seiten 601–610, New York, NY, USA, 2006. ACM.

[ZECH07] Yue Zhang, Serge Egelman, Lorrie Cranor und Jason Hong. Phinding phish: Evaluating anti-phishing tools. In *In Proceedings of the 14th Annual Network and Distributed System Security Symposium (NDSS 2007*, 2007.

DOM-basiertes Cross-Site Scripting im Web: Reise in ein unerforschtes Land*

Ben Stock
Friedrich-Alexander-Universität
91058 Erlangen
ben.stock@cs.fau.de

Sebastian Lekies Martin Johns
SAP AG
76131 Karlsruhe
{sebastian.lekies|martin.johns}@sap.com

Abstract: Cross-site Scripting (XSS) ist eine weit verbreitete Verwundbarkeitsklasse in Web-Anwendungen und kann sowohl von server-seitigem als auch von client-seitigem Code verursacht werden. Allerdings wird XSS primär als ein server-seitiges Problem wahrgenommen, motiviert durch das Offenlegen von zahlreichen entsprechenden XSS-Schwächen. In den letzten Jahren jedoch kann eine zunehmende Verlagerung von Anwendungslogik in den Browser beobachtet werden eine Entwicklung die im Rahmen des sogenannten Web 2.0 begonnen hat. Dies legt die Vermutung nahe, dass auch client-seitiges XSS an Bedeutung gewinnen könnte. In diesem Beitrag stellen wir eine umfassende Studie vor, in der wir, mittels eines voll-automatisierten Ansatzes, die führenden 5000 Webseiten des Alexa Indexes auf DOM-basiertes XSS untersucht haben. Im Rahmen dieser Studie, konnten wir 6.167 derartige Verwundbarkeiten identifizieren, die sich auf 480 der untersuchten Anwendungen verteilen.

1 Einleitung und Motivation

Der Begriff *Cross-Site Scripting (XSS)* beschreibt eine Klasse von Injektionsangriffen, bei der es dem Angreifer möglich ist, eigenen HTML- und JavaScript-Code in die Seiten der angegriffenen Anwendung einzuschleusen. In ihrem Ranking der kritischsten Sicherheitsprobleme von Web-Anwendungen platziert das Open Web Application Security Project (OWASP) XSS auf Platz drei [Ope13] und das Sicherheitsunternehmen Whitehat Security berichtete in der diesjährigen Ausgabe ihres Sicherheitsreports, dass 43% aller Schwachstellen in Web-Applikationen in die Kategorie Cross-Site Scripting einzuordnen waren [Whi13a]. In der allgemeinen Literatur wird XSS meist als server-seitiges Problem behandelt [Eck04]. Dementsprechend beschränken sich die etablierten Gegenmaßnahmen meist auf server-seitige Filterung der potenziell unvertrauenswürdigen Nutzereingaben und der Bereinigung des finalen HTML/JavaScript-Codes über Output Sanitization, bevor dieser an den Web-Browser geschickt wird.

Derweil führt die Unterklasse des DOM-basierten XSS [Kle05], in dem die XSS-Lücke durch rein client-seitigen JavaScript-Code verursacht wird, ein Schattendasein, bedingt durch eine deutlich kleinere Angriffsoberfläche: Sowohl die Menge der client-seitigen

*Technische Grundlage dieses Beitrags ist die englischsprachige Verffentlichung "25 Million Flows Later - Large-scale Detection of DOM-based XSS" [LSJ13]. Im Vergleich bietet dieser Beitrag eine erweiterte Analyse der gesammelten Daten (siehe Abschnitt 5), die in dieser Form im Originalartikel fehlt.

Quellen von Daten, die vom Angreifer kontrolliert werden können, wie auch die Menge an potenziell unsicheren Sprachkonstrukten, die zu XSS Lücken führen, sind ungleich kleiner als auf der Serverseite.

Bis dato gab daher noch keine Studie, welche das Vorkommen dieser Art von Schwachstellen systematisch untersucht hat. Ziel der Arbeit war es, diesen recht unerforschten Schwachstellentyp strukturiert zu erforschen und mögliche Muster in den Verwundbarkeiten aufzudecken. Um eine groß angelegte Untersuchung dieser Art zu ermöglichen, wurden im Rahmen der Arbeit mehrere Komponenten zur automatisierten Analyse von Webseiten entwickelt. Anschließend wurde diese Infrastruktur genutzt, um die Alexa Top 5000 Webseiten in einem *Shallow Crawl* zu untersuchen. Insgesamt konnten im Rahmen dieser Untersuchung auf 480 der Alexa Top 5000 Webseiten Schwachstellen dieser Art festgestellt werden. DOM-basiertes XSS ist also eine relevante Schwachstelle im Web von heute.

Dieses Papier gliedert sich wie folgt: In Abschnitt 2 werden die technischen Hintergründe zur Same-Origin Policy, Cross-Site Scripting sowie im speziellen DOM-basiertem Cross-Site Scripting vorgestellt. Abschnitt 3 stellt dann die Kern-Komponenten des von uns entwickelten Systems vor, mit dem die Untersuchungen durchgeführt wurden. Im Anschluss werden in Abschnitt 4 die Vorgehensweise und die Ergebnisse der durchgeführten empirischen Studie diskutiert. Abschnitt 5 analysiert dann die Ergebnisse der Studie und zeigt die gewonnenen Erkenntnisse auf. Zuletzt werden verwandte Arbeiten erläutert (Abschnitt 6) und abschließend der wissenschaftliche Beitrag der Arbeit zusammengefasst (Abschnitt 7).

2 Technischer Hintergrund

Im Folgenden werden die Grundlagen zur Same-Origin Policy (SOP) und Cross-Site Scripting als Umgehung der SOP gelegt. Im Anschluss wird speziell auf DOM-basiertes Cross-Site Scripting eingegangen und die Kontext-Sensitivität von Cross-Site Scripting erläutert.

2.1 Die Same-Origin Policy und Cross-Site Scripting

Das grundlegendste, in Browser implementierte Prinzip zur Abgrenzung von Applikation voneinander ist die *Same-Origin Policy* (SOP) [Bar09]. Die SOP schützt Resourcen vor Fremdzugriff und erlaubt den Zugriff nur dann, wenn der *Origin* der beiden interagierenden Resourcen übereinstimmt. Der Origin ist dabei über die Kombination aus Protokoll, Domain und Port definiert.

Cross-Site Scripting beschreibt eine Reihe von Angriffen, die auf das Einschleusen von Script-Code abzielt, um die Same-Origin Policy zu umgehen. Gelingt es einem Angreifer, seinen eigenen JavaScript-Code durch eine Schwachstelle in eine Applikation einzuschleusen, wird dieser Code im Kontext der Applikation im Browser des Opfers ausgeführt. Konkret bedeutet dies, dass der Angreifer damit im Namen des Benutzers mit der Applikation interagieren kann. Das Schadpotenzial solcher Angriffe zeigt das aktuelle Beispiel der Seite `ubuntuforums.org`, von der 1,82 Millionen Benutzerdaten durch Ausnutzen einer Cross-Site Scripting-Schwachstelle gestohlen werden konnten [Whi13b].

Die in der Literatur erwähnten, altbekannten Arten von Cross-Site Scripting sind dabei

das persistente und reflektierte XSS. Bei ersterer Art hat ein Angreifer die Möglichkeit, seinen Schadcode beispielsweise in einer Datenbank zu persistieren, der dann bei jedem zukünftigen Besucher der Applikation ausgeführt wird. Beim reflektierten XSS hingegen entsteht die Verwundbarkeit dadurch, dass ein Teil der Anfrage des Clients in die Seite reflektiert wird. Ein Angreifer kann sein Opfer so in der verwundbaren Applikation auf eine präparierte URL locken, der der Schadcode angehängt ist. Sobald das Opfer diese Seite besucht, wird der Code des Angreifers ausgeführt.

2.2 DOM-basiertes Cross-Site Scripting

Neben den zwei bekannten Arten des Cross-Site Scriptings (XSS) wurde erstmal im Jahr 2005 von Amit Klein der Begriff des *DOM-basierten Cross-Site Scriptings* (DOMXSS) diskutiert. Klein fasste unter diesem Begriff jedwede XSS-Schwachstellen zusammen, welche auf Fehler im client-seitigen Code zurückzuführen sind. Dies grenzte DOMXSS wesentlich von den bereits erwähnten Arten ab, welche durch server-seitige Programmierfehler entstehen. Listing 1 zeigt eine solche Schwachstelle beispielhaft. Sowohl im Internet Explorer als auch in Googles Chrome werden Teile der URL, auf die per `location.href` zugegriffen wird, nicht automatisch enkodiert. Ein Angreifer kann sein Opfer auf eine Seite locken, welche etwa die Form

```
http://domain.de/seite#"></script>SchadCode
```

hat. Da mittels `href` die ganze URL an die entsprechende Stelle im Dokument geschrieben wird, kann der Angreifer hiermit aus dem bestehenden `script` ausbrechen und seinen eigenen Schadcode ausführen. Da dieser im Kontext der verwundbaren Seite ausgeführt wird, kann der Angreifer nun im Namen des Benutzers mit der Applikation interagieren.

Das unterliegende Problem von Cross-Site Scripting lässt sich zu einem Datenfluss-Problem abstrahieren. Vom Angreifer kontrollierbare Daten, welche aus verschiedenen *Quellen* stammen, fließen im Verlauf der Ausführung des Codes in sicherheitskritische *Senken*. Sofern die Daten nicht ausreichend validiert oder enkodiert werden, können solche *Flüsse* für einen XSS-Angriff genutzt werden. Die von uns betrachteten Quellen und Senken entsprachen dabei denen im DOMXSS Wiki aufgelisteten [DHB].

2.3 Kontext-Sensitivität von Cross-Site Scripting

Unter dem Begriff Cross-Site Scripting wird häufig das Einschleusen von Script-Elementen verstanden, obwohl dies nur eine Art eines XSS-Angriffs ist. Werden die eingeschleusten Daten beispielsweise innerhalb von JavaScript-Code in einem Aufruf an `eval` genutzt, muss der Angreifer kein `script`-Element einschleusen. Stattdessen muss er den an `eval`

Listing 1 Beispielhafte Verwundbarkeit

```
document.write('<script src="//werbung.de/werbung.js?referrer='
    + document.location.href +'"></script>');
```

übergebenen Code derart manipulieren, dass der gewünschte Schadcode ausgeführt wird. Wie dies verdeutlicht, sind Verwundbarkeiten bei allen Arten von Cross-Site Scripting kontext-abhängig. Im erstgenannten Fall muss der Angreifer seine eingeschleusten Daten derart wählen, dass valides HTML entsteht, wohingegen im zweiten Fall valides JavaScript geschrieben werden muss. Obwohl diese beiden Arten von Senken sehr unterschiedlich sind, lassen sich alle Angriffe wie folgt abstrahieren:

$$\textit{Angriff} := \textit{Ausbruch-Sequenz} \| \textit{Schadcode} \| \textit{Kommentar-Sequenz}$$

Hierbei steht das Zeichen $\|$ für die Konkatenation zweier Strings. Im Falle eines Angriffs ist der Schadcode beliebig vom Angreifer zu bestimmen. Im Gegensatz dazu ist die Ausbruch-Sequenz hochgradig kontext-abhängig. Hier muss geprüft werden, ob sich, beispielsweise im Falle von HTML, die eingeschleusten Daten innerhalb einer Attribut-Zuweisung befinden. In diesem Fall muss zuerst aus der Zuweisung und anschließend aus der Deklaration des HTML-Knotens an sich ausgebrochen werden. Die Kommentar-Sequenz hingegen kann anhand der Senke – hier wird zwischen HTML- und JavaScript-Kontexten unterschieden – sehr einfach bestimmt werden und bedarf keiner genaueren Analyse der Daten, die in die Senke geschrieben werden.

3 Entwickelte Komponenten

Die von uns im Rahmen dieser Arbeit entwickelte Infrastruktur zur automatischen Erkennung von DOM-basierten Schwachstellen besteht aus drei Komponenten: einem modifizierten Browser, welcher Taint-Tracking unterstützt, einem Programm zur automatischen Generierung von prototypischen Exploits sowie einem Crawling-Modul, welches als Browser-Erweiterung implementiert wurde. Im Folgenden stellen wir diese Komponenten vor.

3.1 Erweiterung von Chromium um Taint-Tracking

Um die Erkennung von Datenflüssen aus Quellen in sicherheitskritische Senken zu ermöglichen, entschieden wir uns für die Implementierung eines dynamischen Taint-Tracking-Systems in einen bereits bestehenden Browser. Unsere Wahl fiel dabei auf Chromium, dem Open-Source-Gegenstück von Googles bekanntem Browser Chrome [Goo] . Im Vergleich zu einer automatisierten Test-Suite wie HTMLUnit [HTM] implementiert Chromium bereits alle aktuellen HTML5-APIs und wird in Bezug auf weitere Neuerungen zeitnah aktualisiert.

Die wichtigsten Anforderungen an das von uns implementierte Taint-Tracking waren neben dem korrekten Markieren aller Daten aus relevanten Quellen auch die ordnungsgemäße Weitergabe der Informationen bei Transformation der Strings wie beispielsweise dem Verbinden zweier Strings oder der Extraktion einzelner Buchstaben. Dazu wurden die Implementierungen von Zeichenketten in der V8 JavaScript-Engine sowie die in der Webkit-Rendering-Engine so erweitert, dass diese zusätzlich Taint-Information speichern konnten. Die Änderungen erlaubten es uns, für jeden Buchstaben einer Zeichenkette die Quelle zu speichern. Im Kontext von DOM-basiertem XSS betrachten wir lediglich 14 verschiedene Quellen (darunter alle Teile der URL, Cookies, postMessages und Web Storages). Daher war für die Speicherung der Quellidentifizierer lediglich ein Byte pro Buch-

stabe von Nöten. Alle String-Operationen wurden entsprechend modifiziert, um eine Weitergabe des Taints sicher zu stellen. Nach den Änderungen der String-Implementierungen wurden nun alle Quellen im Sinne von DOM-basiertem XSS so modifiziert, dass diese Taint-Informationen entsprechend angehängt wurden. Analog dazu wurden ebenfalls alle Senken erweitert, um bei einem Fluss eines getainteten Strings eine Meldung an unsere Browser-Erweiterung zu senden. Die Funktion der Erweiterung wird im Laufe dieses Kapitels näher erläutert.

Wie bereits erläutert sind im Sinne von DOM-basiertem Cross-Site-Scripting lediglich 14 Quellen relevant, welche schon durch vier Bits abgebildet werden können. JavaScript bietet Progammieren die Möglichkeit, bereits fest eingebaute Funktionen zum Kodieren von Zeichenketten zu nutzen – *escape*, *encodeURI* und *encodeURIComponent*. Jede dieser Funktionen kann verhindern, dass eine potenzielle Schwachstelle ausnutzbar ist, wenn sie im jeweils richtigen Kontext genutzt wird. Um zusätzlich zu den Quellen eines jeden Buchstabens auch feststellen zu können, ob eine der vorgenannten Funktionen genutzt wurde, um den String zu maskieren, wurden die drei niederwertigsten der verbleibenden Bits genutzt, um die Maskierungen zu vermerken.

3.2 Browser Extension

Nachdem unser Prototyp nun in der Lage war, Taint-Informationen anzuhängen, weiterzureichen und abschließend auch einen Flu in eine Senke zu melden, entwickelten wir eine Chrome Extension [Goo12], deren Aufgabe es war, die Taint-Informationen aufzubereiten and an unser zentrales Backend zu senden. Neben dieser Funktion wurde auch ein Crawler in der Erweiterung implementiert, um die Datengewinnung in groß angelegter, automatisierter Weise möglich zu machen. Die Funktionsweise des gesamten Systems inklusive erweitertem Browser zeigt Abbildung 1. Das Background-Script steuert dabei die einzelnen Tabs, in denen jeweils die zu analysierende Seite zusammen mit dem Content- und User-Script geladen wird. Das Content-Script hat gemeinsam mit dem User-Script die Aufgabe, alle Links einer gerade analysierten Seite einzusammeln und mit Hilfe des Background-Scripts an das Backend zu versenden. Ebenfalls ist es Aufgabe dieser Komponenten, die gemeldeten Flüsse an das Backend weiterzuleiten.

3.3 Automatische Generierung von Exploits

Mit den bisher vorgestellten Komponenten waren wir in der Lage, im großen Maße Daten über potenziell verwundbare Flüsse zu sammeln. Da allerdings neben den vom Browser zur Verfügung gestellten Kodierungsfunktionen häufig auch vom Webseiten-Entwickeler entworfene Filter-Funktionen genutzt werden, ist ein potenziell verwundbarer Fluss nicht gleichbedeutend mit einer Schwachstelle. Um dennoch zu verifizieren, ob es sich bei einem Fluss um eine Schwachstelle handelt, wurde im Rahmen dieser Arbeit daher ein Programm entwickelt, welches automatisch für einen gegebenen Fluss einen prototypischen Exploit generieren kann.

Wie bereits anfangs erläutert, ist die Generierung eines validen Exploits stark vom Kontext abhängig. Dementsprechend wurden im Rahmen der Arbeit eigene Komponenten für das Erzeugen von Ausbruch-Sequenzen für JavaScript- und HTML-Kontexte entwickelt. Im Falle von HTML bestehen diese Sequenzen typischerweise aus Zeichen, welche Attribut-

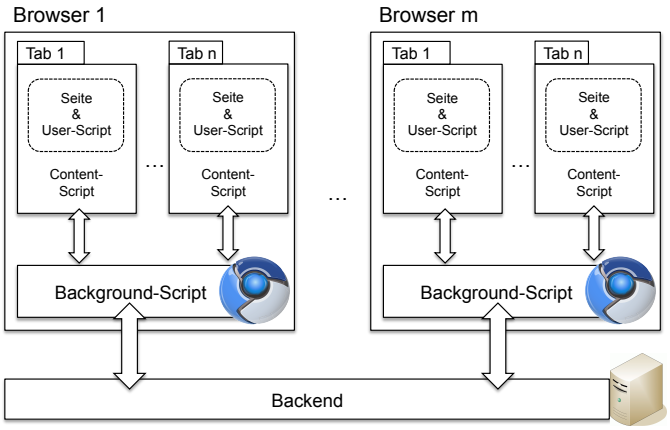


Abbildung 1: Übersicht der Crawler-Infrastruktur

Zuweisungen beenden, öffnende Deklarationen von HTML-Knoten schließen und gegebenfalls schließende HTML-Knoten anhängen. So interpretiert ein Browser beispielsweise jedwede Daten, die sich zwischen einem öffnenden und schließenden `iframe` befinden, nur dann, wenn keine Frames unterstützt werden. In diesen Fällen wurde vom Exploit-Generator eine entsprechende Ausbruch-Sequenz inklusive schließendem `iframe` generiert, um Ausführung des von uns gewünschten Codes zu garantieren.

Im Gegensatz zur Generierung von Ausbruch-Sequenzen für HTML-Kontexte, ist das Erzeugen von Ausbruch-Sequenzen für JavaScript komplizierter. Insbesondere ist hierbei zu beachten, dass im Falle eines Aufrufs an `eval` nur dann überhaupt Code ausgeführt wird, wenn der übergebene String syntaktisch korrektes JavaScript ist. Im ersten Schritt zur Generierung eines validen JavaScript-Exploits wird daher der abstrakte Syntaxbaum der Strings erzeugt. Innerhalb des so entstehenden Baumes wird dann die Position bestimmt, an denen getaintete Werte vorkommen. Ausgehend von dem gefundenden Ast wird der Baum nach oben durchlaufen und jeweils die Äste syntaktisch korrekt beendet. Listing 3 zeigt beispielhaft einen solchen Baum, generiert für den in Listing 2 gezeigten JavaScript-Code. Um einen validen Exploit zu erzeugen, muss zuerst das String-Literal mit `"` abgeschlossen werden. Anschließend wird die Deklaration der Variable `x` mittels `;` sowie der umgebende Block mit } beendet. Die entsprechende Ausbruch-Sequenz, auf die jetzt vom Angreifer gewählter Schadcode folgen kann, lautet also `";}`. In unserem Exploit-Generator wurden automatisch diese Ausbruchs-Sequenzen generiert, die im Anschluss

Listing 2 JavaScript-Auszug

```
var code =  'function test(){' +
      'var x = "' + location.href + '";' //innerhalb von test
    + '}'; //globaler Scope
eval(code);
```

Listing 3 Syntaxbaum zu Listing 2

```
FunctionDeclaration
  Identifier : test
  FunctionConstructor
    Identifier : test
    Block
      Declaration
        Identifier : x
        StringLiteral : "http://example.org"
```

im globalen Scope die Ausführung vom eingeschleusten JavaScript-Code ermöglichen.

4 Empirische Studie

Im Anschluss an die Implementierung der vorgestellten Komponenten führten wir eine empirische Studie durch, um die Anzahl der Verwundbarkeit, die sich auf DOM-basiertes Cross-Site Scripting zurückführen lassen, zu bestimmen. Dazu führten wir mit Hilfe unserer Crawler-Infrastruktur einen *Shallow Crawl* – das Besuchen der Startseite sowie aller auf der Startseite gefundenen Unterseiten – der Alexa Top 5000 Webseiten durch. Im Folgenden präsentieren wir die Vorgehensweise und Ergebnisse dieser Studie.

Für das Sammeln der potenziell verwundbaren Flüsse setzten wir insgesamt fünf PCs ein, welche jeweils mit vier Tabs die Alexa Top 5000 auf die vorgenannte Art besuchten. Im Zeitraum von fünf Tagen besuchten die Crawler dabei insgesamt 504.275 Webseiten. Im Durchschnitt beinhalteten diese Seite jeweils 8,64 Frames, wodurch sich die Anzahl der insgesamt untersuchten Dokumente auf insgesamt 4.538.031 belief. Auf diesen Seiten wurden insgesamt 24.474.306 Datenflüsse registriert und an das Backend gemeldet.

Aufgrund der überraschend hohen Anzahl an potenziell verwundbaren Flüsse und um die automatisierte Validierung in sinnvoller Zeit sicherzustellen, entschieden wir uns, nur für ausgewählte Flüsse prototypische Exploits zu generieren. Die Kriterien für die Auswahl waren dabei:

- Die Senke erlaubt direkte Ausführung von JavaScript. Dementsprechend wurden alle Flüsse in Quellen wie Cookies, WebStorage oder Attribute von DOM-Knoten ausgegrenzt.
- Die Quelle der Daten kann direkt vom Angreifer kontrolliert werden, muss also entweder Bestandteil der URL oder der Referrer sein.
- Lediglich solche Flüsse sollen analysiert werden, die nicht kodiert sind.

Auf diese Weise wurden insgesamt 313.794 Flüsse identifiziert, für die von unserem Generator prototypische Exploits erzeugt werden konnten. In den so entstandenen und zu besuchenden URLs mit angehängtem “Schadcode” befanden sich allerdings viele Duplikate, was darauf zurück zu führen ist, dass einzelne Dokumente oftmals den gleichen Script-Code an mehreren Stellen nutzen. Dementsprechend wird ein potenziell unsicherer Fluss mehrfach registriert. Nach Entfernung der Duplikate verblieben 181.238 eindeutige URLs,

die im Anschluss zur Verifizierung an unsere Crawling-Infrastruktur zurückgegeben wurden.

Um die Validierung der Schwachstelle automatisieren zu können, erweiterten wir die Chrome Extension um eine Funktion zur Exploit-Verifikation. Diese JavaScript-Funktion, welche über das Content-Script in jede besuchte Seite eingebunden wurde, erhielt als Parameter die eindeutige Identifikationsnummer eines Flusses und vermeldete diesen mit zusätzlichen Information wie der gerade geladenen URL an unser Backend. Somit konnten wir mit Hilfe des Exploit-Generators URLs mit prototypischen Exploits generieren, welche bei erfolgreichem Ausnutzen einer Schwachstelle `report(ID)` aufriefen. Die `ID` steht dabei für den numerischen Identifizierer eines Flusses in unserer Datenbank.

Im Anschluss wurden die Crawler erneut gestartet, um die generierten Exploit-Prototypen zu verifizieren. Auf 69,987 der 181.238 URLs, die von den Crawlern besucht wurden, wurde die `report`-Funktion erfolgreich aufgerufen und somit eine Schwachstelle nachgewiesen. Wie bereits erwähnt, wurden im Rahmen unseres initialen Crawls diverse Unterseiten der Alexa Top 5000 besucht. Viele der von uns besuchten Seiten setzten dabei Content-Management-Systeme ein, was den Verdacht nahe legt, dass Schwachstellen potenziell auf allen Unterseiten vorhandenen sind. Um ein genaueres Bild über die tatsächlich Anzahl der einzigartigen Verwundbarkeiten zu erhalten, wandten wir auf die URLs, die als verwundbar gemeldet wurden, ein Eindeutigkeitskriterium an. Dieses Kriterium bestand dabei aus dem Tupel

$$\{\text{Domain}, \text{Ausbruch-Sequenz}, \text{Code-Typ}, \text{Position im Code}\}\,.$$

Der *Code-Typ* unterscheidet dabei, ob der Script-Code *inline*, *eval* oder *extern* gespeichert war. Inline-Code meint dabei solchen JavaScript-Code, der direkt mittels `script`-Knoten innerhalb des Dokuments eingebettet ist, wohingegegen eval solchen Code meint, der innerhalb eines Aufrufs an `eval` ausgeführt wurde. Externer Code ist dabei Code, welcher per `script`-Element mit `src`-Attribut aus einer externen Datei nachgeladen wird. Die relevante Position im Code ist im Falle eines externen Scripts dabei sowohl die Zeile als auch die Position innerhalb der Zeile; für eval und inline Code lediglich die Position innerhalb der Zeile.

Nach Anwendung dieses Kriteriums wurde die Menge der einzigartigen Schwachstellen auf 8.163 auf insgesamt 701 Domains reduziert. Da jedoch im Rahmen des Crawls auch von solchen Seiten Frames nachgeladen wurden, die ausserhalb der Top 5000 lagen, analysierten wir die Daten erneut in Hinblick auf den Alexa-Rang. Dabei stellte sich heraus, dass sich die Anzahl der eindeutigen Verwundbarkeiten in den Alexa Top 5000 auf 6.167 (auf 480 Domains) belief. In Bezug auf die Anzahl der besuchten Domains bedeutet dies, dass sich auf 9,6% der 5000 meistbesuchten Webseiten mindestens eine DOMXSS-Schwachstelle befindet. Unter den verwundbaren Seiten befanden sich neben diversen Banken und einem sozialen Netzwerk auch verschiedene Seiten von Behörden sowie zwei Anti-Viren-Hersteller.

5 Erkenntnisse der Studie

In diesem Kapitel diskutieren wir die Erkenntnisse, die wir aus unserer durchgeführten Studie gewinnen konnten. Dabei geben wir eine detaillierte Analyse der Verwundbarkeiten

in Hinblick auf die genutzten Senken, die Herkunft des verwundbaren Script-Codes sowie einer allgemeinen Analyse der Anwendung von Kodierungs-Funktionen auf Daten der für DOMXSS relevanten Quellen.

5.1 Analyse der Senken

Tabelle 1 zeigt die Verteilung der Flüsse, die insgesamt für die Senken gemessen wurden im Vergleich zu Flüssen in diese Senken, welche verwundbar waren. Zudem bildet sie ab, wie die prozentuale Verteilung der Schwachstellen auf die drei von uns untersuchten Senken war. Auffällig ist dabei, dass offenbar in Fällen, in denen ein Datenfluss in `eval` endet, dieser zu einem größeren Anteil kodiert ist, wohingegen insbesondere im Falle von `innerHTML` in fast einem Viertel der Fälle gänzlich auf Kodierung verzichtet wird. Von allen Flüssen, die anhand unserer im Crawl gesammelten Daten als potenziell gefährlich erkannt wurden, konnten im Falle der HTML-Senken `innerHTML` und `document.write` bei jedem viertem Fluss eine Verwundbarkeit nachgewiesen werden. Im Gegensatz dazu konnten Flüsse, die in `eval` endeten, nur in 4% aller Fälle ausgenutzt werden. Dies legt die Vermutung nahe, dass Programmierer im Umgang mit `eval` mehr Gefahrenpotenzial sehen als mit HTML-Senken und dementsprechend häufiger die genutzten Daten kodieren bzw. validieren.

5.2 Herkunft des verwundbaren Codes

Der zweite Fokus bei der Auswertung der gewonnen Daten lag auf der Analyse der Verwundbarkeiten in Bezug auf ihre Herkunft. Häufig binden Webseiten Script-Inhalte – insbesondere im Falle von Werbung – von fremden Servern ein. Obwohl dieser Inhalt faktisch nicht den gleichen Origin im Sinne der Same-Origin Policy hat, wird der Script-Code dennoch im Kontext der Seite ausgeführt, welche den Inhalt eingebunden hat. Dies bedeutet, dass eine Verwundbarkeit in Drittanbieter-Code in dem Moment, in dem eine Seite diesen Code einbindet, zu einer Schwachstelle in dieser Seite führt. Unsere Untersuchung ergab, dass 13,09% aller Schwachstellen auf solchen, von Dritten bereitgestellten, Code zurückgingen (*cross-domain extern*). Der größte Anteil an Schwachstelle fiel mit 79,64% jedoch auf Scripte, die zwar aus einer separaten Datei nachgeladen wurden, deren Quelle allerdings die gleiche Domain wie die Seite selbst war (*same-domain extern*). Einen vergleichsweise kleinen Anteil machten mit 3,81% Inline-Scripte aus. Der Rest der Schwachstellen entstand durch Code, welcher mittels `eval` erzeugt und ausgeführt wurde. Aufgrund der Funktionsweise der JavaScript-Engine war es bei diesen verbleibenden Schwachstellen aber nicht möglich, die genaue Position (inline, cross-domain extern oder

Senke	Flüsse	unkodiert	ausnutzbar[1]	ausgenutzt[2]	eindeutig
eval	1.728.872	81.949	47.925	1.925 (4,02%)	319
document.write	2.587.206	291.837	170.793	44.049 (25,79%)	7.170
innerHTML	631.286	145.322	95.076	24.013 (25,26%)	674

[1] aus Quellen, die direkt vom Angreifer kontrolliert werden können
[2] Prozentangaben beziehen sich auf ausnutzbare Flüsse nach Spalte 4

Tabelle 1: Untersuchte Senken mit gesamten und exploitbaren Flüssen

same-domain extern) zu bestimmen.

5.3 Anwendung von Kodierungsfunktionen

Insgesamt zeichnete sich außerdem bei der Auswertung der Daten ab, dass es erhebliche Unterschiede in der Verarbeitung von Daten aus den unterschiedlichen Quellen gab. So wurden Daten aus der URL immerhin in fast 65% aller Fälle kodiert, bevor sie in einer der relevanten Senken endeten – beim Referrer, also der URL der Seite, über die der Benutzer auf die aktuelle Seite gekommen ist, wurden die Daten sogar in fast 84% aller Fälle kodiert. Im krassen Gegensatz dazu stehen Daten aus postMessages, welche nur in 1,57% aller Fälle kodiert waren. Die postMessage-API, welche im Zuge von HTML5 in Browser eingefügt wurde, erlaubt es Applikationen innerhalb eines Browsers mit Hilfe von Nachrichten zu kommunizieren. So lassen sich damit etwa zwischen einem geöffneten Popup und der öffnenden Seite Nachrichten auch über Origin-Grenzen hinweg austauschen. Die API erlaubt es der empfangenden Seite, den Absender der Nachricht zu überprüfen. Sofern diese Prüfung jedoch nicht korrekt oder gar nicht durchgeführt wird, kann ein Angreifer eine beliebige Nachricht an die empfangende Seite schicken. Obwohl wir in unserer Arbeit keine Verifikation von derartigen Schwachstellen durchgeführt haben, liegt der Verdacht – auch aufgrund von verwandten Arbeiten [SS13] – nahe, dass mehrere der von uns untersuchten Webseiten verwundbar sind.

Im Zuge unserer Untersuchung fanden wir zudem bei händischen Tests mit einer generierten Exploit-Payload eine Schwachstelle, in der zuerst lediglich ein Fluss in ein Cookie stattfand. Ein solcher Fluss ist noch nicht notwendigerweise verwundbar, allerdings wurde im konkreten Fall beim zweiten Laden der Seite dieser Cookie-Wert genutzt, um einen String zusammen zu setzen, welcher anschließend an `eval` übergeben wurde. Dementsprechend konnte beim zweiten Laden der Seite erfolgreich der Exploit ausgelöst werden. Dieses Beispiel verdeutlicht auch die Wichtigkeit der Benutzung von Kodierungsfunktionen, selbst wenn die zu kodierenden Daten nicht direkt in eine solche Senke, die sofortige Ausführung von Schadcode ermöglicht, fließen.

6 Verwandte Arbeiten

Das Konzept einer um Taint-Tracking erweiterten Browsing-Engine wurde erstmals bei DOMinator verfolgt [Di 12]. DOMinator bietet allerdings im Gegensatz zum von uns entwickelten Prototypen nicht die Möglichkeit, genaue Taint-Information zu jedem Zeichen eines Strings zu erhalten. Dementsprechend kann dieser auch nicht als Basis des von uns entwickelten, automatischen Exploit-Generators genutzt werden. Konzeptionell am nächsten zum vorgestellten Ansatz ist FLAX [SHPS10], welches ebenfalls byte-genaues Taint-Tracking nutzt, um unsichere Datenflüsse in JavaScript zu erkennen. Der große Unterschied besteht dabei allerdings darin, dass FLAX Teile des Programm-Codes in eine vereinfachte Variante von JavaScript überführt und anschließend auf den Teilresultaten nach potenziellen Verwundbarkeiten sucht. Im Gegensatz dazu erlaubt unser Ansatz eine Benutzung der gesamten JavaScript-Funktionalität. Criscione [Cri13] stellte im April 2013 ein von Google entwickeltes Systems vor, welches auf automatisierte Art und Weise nach Verwundbarkeiten sucht. Ähnlich zu dem von uns vorgestellten Ansatz werden hierbei instrumentarisierte Browser genutzt, wobei statt Taint-Tracking mit kontext-sensitiver

Exploit-Generierung Blackbox-Fuzzing eingesetzt wird. In Bezug auf die groß angelegte Analyse von Cross-Site Scripting-Verwundbarkeiten legten 2009 Yue und Wang [YW09] erste Grundlagen, in dem sie unsichere Programmierpraktiken von JavaScript untersuchten. Der Fokus bei dieser Arbeit lag jedoch nur auf der statitischen Auswertung von potenziell verwundbarem Code und nicht auf Verifikation der Schwachstellen. Richards et al. [RHBV11] untersuchten 2011 die unsichere Benutzung von `eval` und zeigten Wege auf, die gleiche Funktionalität auf sicherem Wege zu implementieren. Son und Shmatikov [SS13] untersuchten kürzlich, wie viele der Alexa Top 10.000 Webseiten bereits die HTML5-API postMessage nutzen. Dabei stellten sie fest, dass diese neue API bereits auf 22,5% der Top 10.000 Webseiten genutzt wurde, wobei insgesamt 84 Webseiten aufgrund fehlender Prüfung des Nachrichten-Absenders anfällig für Cross-Site Scripting-Angriffe waren.

7 Zusammenfassung

In diesem Beitrag stellten wir die Ergebnisse einer umfassend angelegten praktischen Studie vor, die sich mit dem Vorkommen von DOM-basiertem Cross-site Scripting befasst und die erste groß angelegte Untersuchung dieser Art von Schwachstellen darstellt.

Die technische Kernkomponente unserer Infrastruktur ist ein erweiterter Chromium-Browser, welcher mittels einer Chrome Extension instrumentarisiert wurde, um vollautomatisch Daten zu potentiell unsicheren Flüssen zu sammeln. Diese Daten wurden anschließend als Eingabe für einen Exploit-Generator genutzt, welcher prototypische Testfälle für die möglicherweise verwundbaren Flüsse generierte. Mit diesem System konnten wir auf 480 der Alexa Top 5000 Domains DOMXSS-Schwachstelle identifizieren.

Wie wir in Abschnitt 5 aufzeigen, erlaubt die Studie erste Einblicke in Muster von Programmierfehlern, die zur Entstehung der Schwachstellen beitragen. Das Hauptergebnis der Studie ist die Erkenntnis, dass DOM-basiertes XSS ein signifikantes Problem ist, das in vielen produktiven Webseiten auftritt und augenscheinlich von den aktuell verwendeten Methoden der sicheren Web-Programmierung nicht hinreichend verhindert wird. Dem zu Folge ist es notwendig, dass in der Zukunft die Ursachen dieser Verwundbarkeitsklasse verstärkt untersucht werden, um wirkungsvolle Gegenmanahmen entwickeln zu können.

Danksagungen

Wir danken Felix Freiling für hilfreiche Kommentare zu einer vorherigen Version dieses Textes. Zudem danken wir den anonymen Reviewern für die hilfreichen Kommentare.

Literatur

[Bar09] Adam Barth. The Web Origin Concept, November 2009.

[Cri13] Claudio Criscione. Drinking the Ocean - Finding XSS at Google Scale. Talk at the Google Test Automation Conference, (GTAC'13), `http://goo.gl/8qqHA`, April 2013.

[DHB] Stefano Di Paola, Mario Heiderich und Frederik Braun. DOM XSS Test Cases Wiki Cheatsheet Project. [online] `https://code.google.com/p/domxsswiki/wiki/Introduction`, abgerufen am 9.12.2013.

[Di 12] Stefano Di Paola. DominatorPro: Securing Next Generation of Web Applications. [software], https://dominator.mindedsecurity.com/, 2012.

[Eck04] Claudia Eckert. *IT-Sicherheit*. Oldenbourg, Muenchen [u.a.], 3., überarb. und erw. aufl. Auflage, 2004.

[Goo] Google. Google Chrome. [online], https://www.google.com/intl/de/chrome/browser/, abgerufen am 9.12.2013.

[Goo12] Google Developers. Chrome Extensions - Developer's Guide. [online], http://developer.chrome.com/extensions/devguide.html, abgerufen am 9.12.2013, 2012.

[HTM] HTMLUnit Development Team. HtmlUnit Webseite.

[Kle05] Amit Klein. DOM based cross site scripting or XSS of the third kind. *Web Application Security Consortium, Articles*, 4, 2005.

[LSJ13] Sebastian Lekies, Ben Stock und Martin Johns. 25 million flows later: large-scale detection of DOM-based XSS. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, Seiten 1193–1204. ACM, 2013.

[Ope13] Open Web Application Security Project. OWASP Top 10 - 2013, Oktober 2013.

[RHBV11] Gregor Richards, Christian Hammer, Brian Burg und Jan Vitek. The Eval That Men Do - A Large-Scale Study of the Use of Eval in JavaScript Applications. In Mira Mezini, Hrsg., *ECOOP*, Jgg. 6813 of *Lecture Notes in Computer Science*, Seiten 52–78. Springer, 2011.

[SHPS10] Prateek Saxena, Steve Hanna, Pongsin Poosankam und Dawn Song. FLAX: Systematic Discovery of Client-side Validation Vulnerabilities in Rich Web Applications. In *NDSS*. The Internet Society, 2010.

[SS13] Sooel Son und Vitaly Shmatikov. The Postman Always Rings Twice: Attacking and Defending postMessage in HTML5 Websites. In *Network and Distributed System Security Symposium (NDSS'13)*, 2013.

[Whi13a] Whitehat Security. Website security statistics report, Mai 2013.

[Whi13b] Zack Whittaker. Ubuntu forums hacked; 1.82M logins, email addresses stolen. Online-Artikel, Juli 2013.

[YW09] Chuan Yue und Haining Wang. Characterizing insecure javascript practices on the web. In Juan Quemada, Gonzalo León, Yoëlle S. Maarek und Wolfgang Nejdl, Hrsg., *WWW*, Seiten 961–970. ACM, 2009.

On the Security of Holder-of-Key Single Sign-On

Andreas Mayer[1], Vladislav Mladenov*[2], and Jörg Schwenk[2]

[1]Adolf Würth GmbH & Co. KG, Künzelsau-Gaisbach, Germany
andreas.mayer@wuerth.com
[2]Horst Görtz Institute, Ruhr-University Bochum, Germany
{vladislav.mladenov, joerg.schwenk}@rub.de

Abstract: Web Single Sign-On (SSO) is a valuable point of attack because it provides access to multiple resources once a user has initially authenticated. Therefore, the security of Web SSO is crucial. In this context, the SAML-based Holder-of-Key (HoK) SSO Profile is a cryptographically strong authentication protocol that is used in highly critical scenarios. We show that HoK is susceptible to a previously published attack by Armando et al. [ACC+11] that combines logical flaws with cross-site scripting. To fix this vulnerability, we propose to enhance HoK and call our novel approach HoK+. We have implemented HoK+ in the popular open source framework SimpleSAMLphp.

1 Introduction

Today's Internet users are forced to register to each website individually and have to manage a plethora of accounts and passwords as part of their daily job. This aspect is not only cumbersome but also seriously insecure, as users frequently choose weak (easy to remember) passwords and/or reuse them on several websites. Furthermore, each website has to reinvent the wheel by building and operating another stand-alone authentication solution and thus suffers from high user management costs.

Web Single Sign-On (SSO), as a subset of identity and access management, was proposed to tackle the described usability, security, and management issues. With SSO, a user authenticates *once* to a trusted third party, called Identity Provider (IdP), and subsequently gains access to all federated websites (i.e. Service Providers) he/she is entitled to – without being prompted with another login dialog.

Nowadays, Web SSO solutions are wide-spread and their importance still continues to grow. In this context, the Security Assertion Markup Language (SAML) [CKPM05] is a flexible and open XML standard for exchanging authentication and authorization statements. Since its invention in 2001, SAML has become the dominant technology for enterprise Web SSO. SAML is also used in research, education, and e-Government scenarios.

* The author was supported by the SkIDentity project of the German Federal Ministry of Economics and Technology (BMWi, FKZ: 01MD11030).

In security critical use cases, SAML Holder-of-Key (HoK) Web SSO [KS10] is applied because it fulfills the highest "level of assurance" as defined by the National Institute of Standards (NIST) [The11]. HoK adds strong cryptographic guarantees to the authentication context and enhances the security of SAML assertion and message exchange by using mutual authenticated secure channels. It builds on the TLS protocol which is ubiquitously implemented in all major browsers (including mobile browsers) and web servers. Therefore, maximum compatibility to existing infrastructure and deployments is given.

Contribution. Although, HoK successfully defends against a wide range of attacks, like man-in-the middle (MITM) and man-in-the-browser (MITB), we show that it is still susceptible to a previously discovered attack by Armando et al. [ACC+11]. To mitigate this vulnerability, we propose to extend HoK and use the strong cryptographic binding in a more holistic approach. We call this countermeasure HoK+. In order to demonstrate the practical feasibility, we have implemented a proof-of-concept in the popular open source framework SimpleSAMLphp [Sim14].

Outline. Related work is given in Section 2. The following section will introduce Web SSO, SAML, and cookie theft. HoK Web SSO is explained in Section 4. The RelayState Spoofing attack applied on HoK is presented in Section 5. Our countermeasure HoK+ along with the implementation is presented in Section 6. We conclude in Section 7.

2 Related Work

Single Sign-On. The Security Assertion Markup Language (SAML) was developed for the secure exchange of XML-based messages and is mostly applied within federated identity management. The most widespread field of use of SAML is Web SSO. Unfortunately, a diversity of attacks have been discovered in the last years.

In 2003, Groß [Gro03] disclosed several adaptive attacks on the SAML Browser Artifact Profile resulting in the interception of the authentication token contained in the URL. Additionally, Groß analyzed the revisited version of SAML [CKPM05], finding further logical flaws and security threats [GP06]. Related vulnerabilities have been analyzed and found in the Liberty Single Sign-On by Pfitzmann and Waidner [PW03]. In 2006 Y. Chan introduced a new parallel session attack to bypass all levels of authentication by exclusively breaking the weakest one among them [Cha06].

In 2008, Armando et al. [ACC+08] built a formal model of the SAML V2.0 Web Browser SSO protocol and analyzed it with the model checker SATMC. The practical evaluation revealed an existing security issue on the SAML interface of Google, allowing a malicious SP to impersonate any user at any Google application. Later on, same authors identified another attack on Google's SAML interface [ACC+11]. They manipulated the `RelayState` parameter in the query string of an HTTP request in order to exploit an existing cross-site scripting (XSS) vulnerability on Google. In this paper, we apply the RelayState Spoofing attack on HoK Web SSO.

Further researches on the security of Web SSO have shown serious flaws resulting in iden-

tity theft and compromising the security of the end-systems. In [SMS+12] the authors published an in-depth analysis of XML Signature. Additionally, they introduced novel techniques to manipulate digitally signed messages, despite the applied integrity protection mechanisms. As a result the authors of the study examined 14 major SAML frameworks and showed that 11 of them had critical XML Signature wrapping flaws allowing the impersonation of any user. Another study regarding the security of SSO systems has been published in 2012 by Wang et al. [WCW12]. The authors examined REST-based authentication protocols like OpenID and found serious logic and implementation flaws resulting in identity theft.

TLS Channel Bindings. In 2009 OASIS[1] standardized the *SAML Holder-of-Key Web Browser SSO Profile* [KS10] using TLS client certificates for strengthening the authentication process. Later on, RFC 5929 [AWZ10] has been published in 2010 and describes three different channel binding types for TLS without using any client certificates.

In 2012, Dietz et al. [DCBW12] have proposed a TLS channel binding called *Origin-Bound Certificates* (OBC) by using a TLS extension. Their approach changes server authenticated TLS channels into mutually authenticated channels by using client certificates created on the fly by the browser. However, their idea requires changes in the TLS protocol, thus all current TLS implementations must be modified.

Recently, Google introduced another TLS extension called *Channel ID* [BH12]. Again, fundamental changes to underlying TLS implementations are required. In summary, the browser creates an additional asymmetric key pair during the TLS handshake and uses the private key to sign all handshake messages up to the `ChangeCipherSpec` message. Subsequently, the signature, along with the public key, is sent encrypted through the TLS channel using the freshly established TLS key material. This is done before finishing the TLS handshake. The browser uses the public key as "Channel ID" that identifies the TLS connection.

In 2013 OASIS published the *SAML Channel Binding Extensions* [Can13] that allows the use of channel bindings in conjunction with SAML. In this manner, the channel bindings of RFC 5929 [AWZ10] can be integrated in all SAML related services (e.g. SSO).

3 Foundations

Web Single Sign-On. In an SSO flow (cf. Figure 1) user U navigates user agent UA (e.g. a browser) and tries to access a restricted resource on SP (1). Thus the user is not authenticated yet, SP generates a token request (2) and redirects UA with the token request to IdP (3,4). In the following step U authenticates himself to IdP (5) according to the supported authentication mechanisms. Subsequently, the security token is issued and sent through UA to SP, where the integrity and authenticity is verified and the content is evaluated (6,7).

[1] https://www.oasis-open.org/

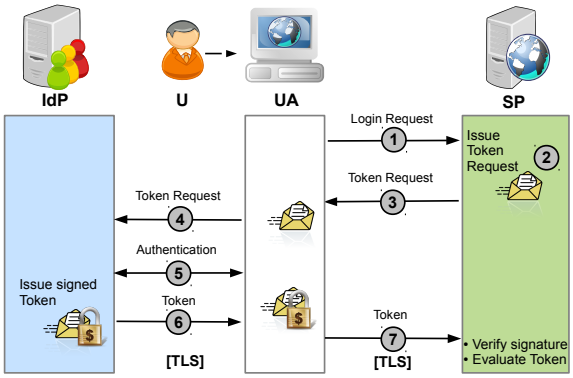


Figure 1: Single Sign-On Overview.

SAML. The Security Assertion Markup Language (SAML) [CKPM05] is a widely-used open XML standard for exchanging authentication and authorization statements about *subjects*. These statements are contained in security tokens called *assertions*. SAML consists of three other building blocks: (1) *protocols* – define how assertions are exchanged between the actors; (2) *bindings* – specify how to embed assertions into transport protocols (e.g. HTTP or SOAP), and (3) *profiles* define the interplay of assertions, protocols, and bindings that are necessary for the needs of a specific use case to be met. The investigated SAML HoK Web Browser SSO Profile [KS10] is such an application scenario.

In order to request an assertion, SAML defines the `<AuthnRequest>` XML message (i.e. token request). Important elements and attributes of this message are: `ID` – a unique and randomly chosen request identifier; `AssertionConsumerServiceURL` (ACS_{URL}) – specifies the endpoint to which IdP must deliver the assertion; `<Issuer>` – the EntityID of SP.

The issued assertion (i.e. security token) contains the following elements and parameters relevant to this paper: `InResponseTo` – must match the `ID` of the `<AuthnRequest>`; `ID` – a unique and randomly chosen assertion identifier; `<Issuer>` – the EntityID of IdP; `<Audience>` – the EntityID of SP; `<SubjectConfirmation>` – the client certificate of UA; `<Subject>` – the EntityID of U;. To assure the integrity and authenticity of the security claims made, the whole assertion *must* be protected by a digital signature (`<Signature>`) which is compliant to the XML Signature standard [ERS+08].

Cookie Theft. The Hypertext Transfer Protocol (HTTP) [FGM+99] is a wide-spread web application protocol transmitting messages between user's browser and web server. Since HTTP is a stateless protocol, without additional mechanisms a user will be forced to re-enter his login information repeatedly, for every HTTP request. HTTP session cookies [Bar11] are applied to solve this problem by making HTTP stateful. In other words, the cookies are a mechanism to make authentication persistent. They are set by the web server, stored in the client's browser and transmitted with every HTTP request to the web server. In this manner a user has to authenticate himself only once in order to get a cookie which represents the session state of an authenticated user.

Unfortunately, HTTP cookies can be stolen by various attacks like eavesdropping the network communication, cross-site scripting (XSS), cross-site request forgery (CSRF), and UI redressing. The theft results in impersonation of the victim by the adversary. To impede cookie theft, two cookie flags are applied: `secure` – defines that cookies are only sent over a secure channel (i.e. TLS); `HTTPOnly` – makes a cookie inaccessible by client-side scripts (e.g. JavaScript). Unfortunately, all existing best-practice cookie theft countermeasures can be bypassed in several ways [Man03, Pal07, BBC11, Hei12].

4 Holder-of-Key Web SSO

The *SAML V2.0 Holder-of-Key (HoK) Web Browser SSO Profile* [KS10] is an OASIS standard based on the browser-based Kerberos scheme `BBKerberos` [GJMS08]. By using mutual authenticated secure channels, HoK adds strong cryptographic binding to the authentication context and enhances the security of the message exchange, i.e. the SAML assertion. HoK builds on the TLS protocol which is ubiquitously implemented in all major browsers (including mobile browsers) and web servers. Therefore, maximum compatibility to existing infrastructures and deployments is given.

In HoK the web server recognizes the browser on basis of a unique (self-signed) client certificate. The browser proves the possession of the client certificate's private key in a mutual authenticated TLS handshake. Any self-signed certificate is sufficient, as neither IdP nor SP are required to validate the trust chain of the certificate. Therefore, no complex and expensive public-key infrastructure (PKI) is needed. The issued assertion is cryptographically bound to the client certificate by including either the certificate itself or a hash of it in the signed assertion. In this manner the assertion can be used only in conjunction with the according private key stored in user's browser.

Figure 2 illustrates the detailed flow of the SAML Holder-of-Key Web Browser SSO Profile. Subsequently, we describe the individual steps:

1. $UA \rightarrow SP$**:** User U navigates its user agent UA to SP and requests a restricted resource R by accessing URI_R.[2] This starts a new SSO protocol run.
2. $SP \rightarrow UA$**:** SP determines that no valid security context (i.e. an active login session) exists. Accordingly, SP issues an authentication request <`AuthnRequest(`ID_1`,` SP`,` ACS_{URL}`)`> and sends it Base64-encoded, along with the RelayState parameter URI_R, as an `HTTP 302` (redirect to IdP) to UA. ID_1 is a fresh random string and SP the identifier of the Service Provider. ACS_{URL} specifies the endpoint to which the assertion must be delivered by IdP.
3. $UA \rightarrow IdP$**:** Triggered by the HTTP redirect, a mutually authenticated TLS connection is established between UA and IdP. Thereby, IdP and UA send to each other their certificates and each proves possession of the corresponding private key within the mutual TLS handshake. Afterwards, the built TLS channel is used to transport <`AuthnRequest(`ID_1`,` SP`,` ACS_{URL}`)`>, along with URI_R, to IdP.

[2] URI_R is called *RelayState* as it preserves and conveys the initial step of a SSO protocol run (i.e. the `URI` of the accessed resource R).

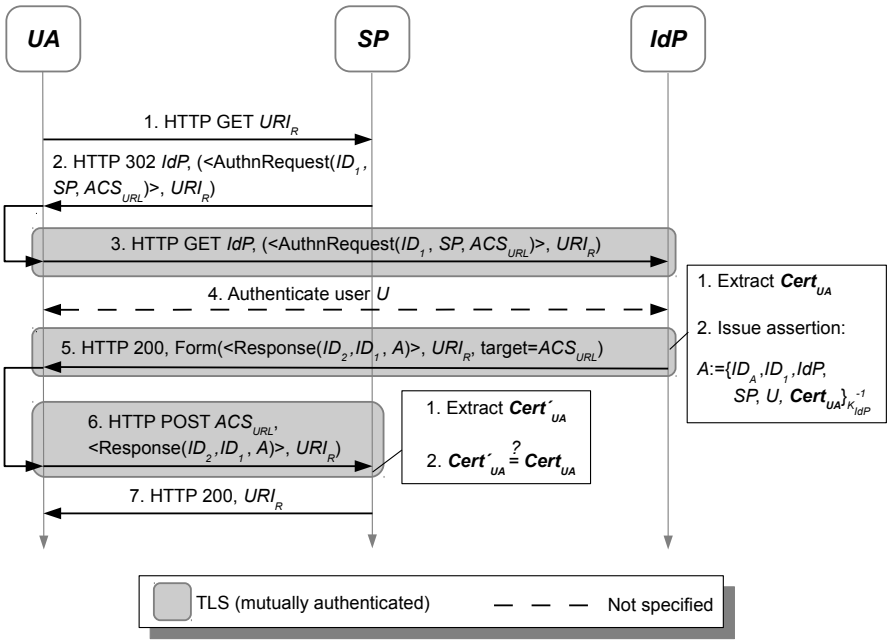


Figure 2: SAML 2.0 Holder-of-Key Web Browser SSO Profile.

4. $UA \leftrightarrow IdP$**:** If the user is not yet authenticated, IdP identifies U by an arbitrary authentication mechanism.
5. $IdP \rightarrow UA$**:** IdP creates an assertion $A := (ID_A, ID_1, IdP, SP, U, Cert_{UA})$, including the unique identifier ID_A and ID_1 from the request, the entity IDs of IdP, SP, the user identity U, and the user certificate $Cert_{UA}$. Subsequently, A is signed with the IdP's private key K^{-1}_{IdP}. The signed assertion A is embedded into a `<Response>` message, together with ID_1 and the fresh response identifier ID_2, and is sent Base64-encoded in an HTML form, along with the `RelayState`=URI_R, to UA.
6. $UA \rightarrow SP$**:** A small JavaScript embedded in the HTML form triggers the forwarding of the assertion A to ACS_{URL} via HTTP POST. Simultaneously, a mutually authenticated TLS channel with SP is built, where UA presents a client certificate $Cert'_{UA}$.
7. $SP \rightarrow UA$**:** SP consumes A, and requires that ID_1 is included as `InResponseTo` attribute in the assertion. A is only valid if the contained $Cert_{UA}$ is equal to $Cert'_{UA}$ from the previous TLS handshake (step 6). SP verifies the XML signature, and authenticates user U resulting in a security context. Finally, SP grants U access to the protected resource R by redirecting U to URI_R.

HoK does not prevent assertion theft in any circumstance (e.g. via XSS). However, stolen assertions are *always* worthless for the adversary, since they are cryptographically bound to the legitimate browser. To successfully attack HoK, the adversary needs knowledge of the private key belonging to the used client certificate. Consequently, the private key is

protected by the browser and/or by the underlying operating system. It is even possible to store the private key on a secure device (e.g. smart card) to protect against malware in untrusted environments (e.g. in kiosk scenarios, where computers are accessible to everyone at public places). Furthermore, HoK protects both TLS connections (between UA and SP as well as between UA and IdP) against MITM and MITB attacks. It is important to note that the presentation of a client certificate in step 1 and 2 (i.e. a mutually authenticated TLS handshake) is *strictly* optional [KS10, p.10].

5 RelayState Spoofing Attack

Armando et al. [ACC+11] have discovered a serious authentication flaw (RelayState Spoofing) in standard Web SSO. In this section, we provide a review of this attack and show that this technique breaks the security of SAML HoK Web SSO.

5.1 Threat Model

We make two assumptions on adversary Adv to launch a successful RelayState Spoofing attack:

1. Adv is able to lure the victim to a malicious website (Adv_{ws}) controlled by him. Since there is no need to read the network traffic, we may assume that UA of the victim always communicates over encrypted TLS channels. Moreover, the victim may only accept communication partners with valid and trusted server certificates.
2. Adv requires an XSS vulnerability for each attacked SP that allows cookie theft.

5.2 Attack Description

The RelayState Spoofing attack combines a logical flaw in the SSO implementation (the `RelayState` parameter URI_R can be changed by Adv, and this parameter will be used in a final redirect triggered by SP) with implementation bugs at the Service Provider SP (an XSS attack can be launched through an HTTP redirect query string parameter). The attack flow is depicted in Figure 3.

In step 2 or 4, Adv injects an XSS attack vector into the parameters of the RelayState $URI_R = BAD_{URI}$. After successful authentication at the honest SP (i.e. after successful verification of the SAML assertion), the maliciously-crafted $URI_R = BAD_{URI}$ is loaded by a browser redirect (step 9), and the XSS attack is automatically executed in the browser resulting in a cookie theft.

Two preconditions must be met for this attack to be successful: (1) injectability of XSS code into URI_R and (2) XSS-vulnerable implementations of SPs. The first precondition

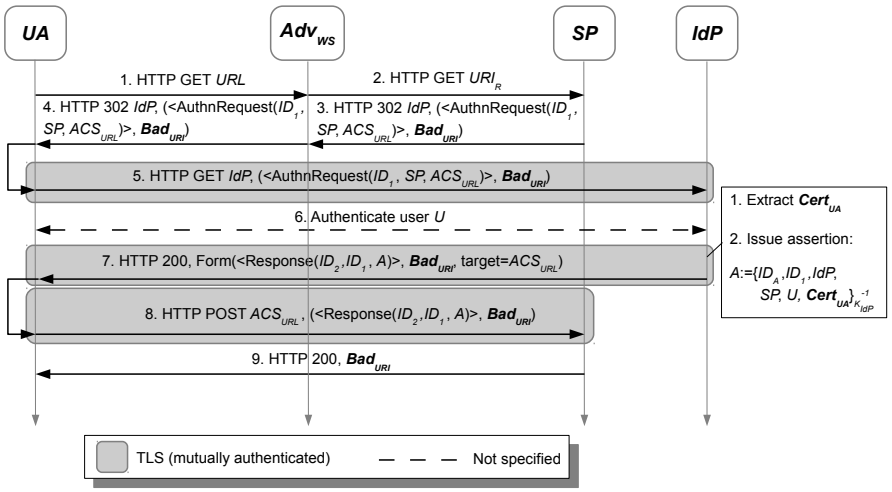


Figure 3: RelayState Spoofing attack on HoK Web SSO.

normally holds in SAML-based SSO scenarios, because URI_R is not part of the XML-based data structures authentication request or assertion, and can thus not be integrity protected by an XML signature. Instead, the SAML standard recommends to protect the integrity of URI_R by a separate signature ([CHK+05], Section 3.4.3). However, [ACC+11] and our own investigations show that this is normally not the case in practice. The second precondition has been shown to be applicable by [WCW12] and [SB12], where numerous implementation bugs for SPs have been documented. Additionally, Armando et al. have presented two successful RelayState Spoofing attacks on Novel Access Manager 3.1 and Google Apps.

By misusing the SSO protocol flow, Adv can ensure that the victim has an authenticated session with the attacked SP, which is a precondition for cookie theft via XSS. Furthermore, Adv can use this attack as launching pad to automatically execute cross-site request forgery (CSRF) [The13] attacks.

An attack related to RelayState Spoofing is *login CSRF* [BJM08], whereby Adv forges a cross-site request to the honest website's login form, resulting in logging the victim into this website as the adversary. This CSRF variant is also applicable to SAML-based SSO.

6 HoK+ Web SSO

Although SAML HoK Web SSO protects against a variety of attacks, it is still susceptible to the RelayState Spoofing attack as shown in the previous section. This is due to the fact, that HoK does not protect the SP's `<AuthnRequest>` against a MITM attack.

To additionally mitigate this severe attack, we propose to enhance HoK and call our novel

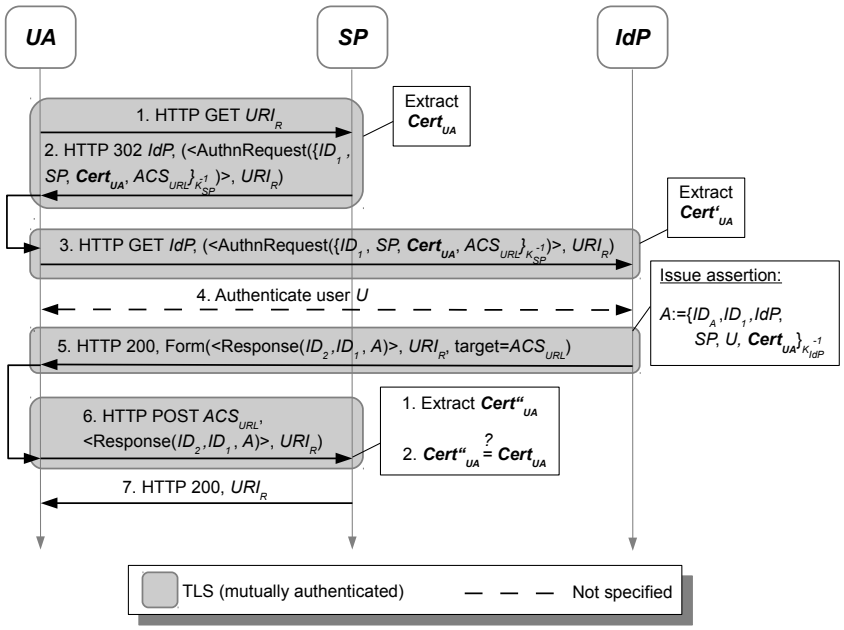


Figure 4: The novel HoK+ SSO Profile.

approach *HoK+*. In summary, HoK+ additionally binds the SP's `<AuthnRequest>` message to the client certificate. Therefore, the *whole* SSO protocol flow is cryptographically linked to the legitimate UA.

6.1 HoK+ Protocol

Figure 4 illustrates the detailed flow of HoK+, which consists of the following steps:

1. $UA \rightarrow SP$**:** User U navigates its user agent UA to SP and requests a restricted resource R by accessing URI_R. This starts a new SSO protocol run. A mutually authenticated TLS connection is established between UA and SP and thereby UA sends its client certificate $Cert_{UA}$ to SP.
2. $SP \rightarrow UA$**:** SP extracts $Cert_{UA}$ from the TLS handshake and issues an authentication request `<AuthnRequest(`ID_1`,` SP`,` $Cert_{UA}$`,` ACS_{URL}`)>`, which is then signed with the SP's private key K_{SP}^{-1}. The `<AuthnRequest>` is sent back to UA, along with URI_R, as HTTP redirect to IdP.
3. $UA \rightarrow IdP$**:** Triggered by the HTTP redirect, a mutual authenticated TLS connection between UA and IdP is established. UA uses this TLS connection to transport `<AuthnRequest>`, along with URI_R, to IdP.
4. $UA \leftrightarrow IdP$**:** IdP verifies the XML signature of the received `<AuthnRequest>`

with SP's public key and then compares $Cert_{UA}$ from the authentication request with $Cert'_{UA}$ of the TLS connection. If they match, IdP authenticates U with an arbitrary method. Otherwise, the protocol is stopped.

5. $IdP \rightarrow UA$**:** IdP creates an assertion $A := (ID_A, ID_1, IdP, SP, U, Cert_{UA})$. Subsequently, A is signed with the IdP's private key K_{IdP}^{-1} and is embedded into a `<Response>` message, together with ID_1 and the fresh response identifier ID_2. Afterwards it is sent Base64-encoded in an HTML form, along with the `RelayState`=URI_R, to UA.
6. $UA \rightarrow SP$**:** A small JavaScript embedded in the HTML form triggers the forwarding of the assertion A to ACS_{URL} via HTTP POST. Simultaneously, a mutually authenticated TLS channel with SP is built, where UA presents a client certificate $Cert''_{UA}$.
7. $SP \rightarrow UA$**:** SP consumes A, and requires that ID_1 is included as `InResponseTo` attribute in the response message and in the assertion. Additionally, A is only valid if the contained $Cert_{UA}$ is equal to $Cert''_{UA}$ from the previous TLS handshake (step 6). SP verifies the XML signature, and authenticates U resulting in a security context. Finally, SP grants U access to the protected resource R by redirecting U to URI_R.

The reason why HoK+ mitigates the RelayState Spoofing attack is that no SAML assertion will be issued by IdP in case of an attack, since the authentication request is bound to the client certificate used by the adversary Adv. Thus we make it impossible for Adv to submit a valid assertion in conjunction with a malicious ACS_{URL} to SP.

6.2 Implementation

In order to demonstrate the feasibility of HoK+, we have implemented it in the popular open source framework SimpleSAMLphp (SSP) [Sim14]. We have chosen SSP because it is known as a fairly secure framework [SMS+12], and because our own penetration tests and source code observations have shown that SSP did not reveal any XSS flaws. Moreover, SSP supports defense-in-depth techniques like `HTTPOnly` and `secure` flag cookies by default.

SSP already supports HoK [MS11]. We added code to create, process, and verify signed HoK+ authentication requests. The enhanced `<AuthnRequest>` is compliant with the SAML standard and conforming to the SAML V2.0 XML schema. The client certificate is added in the same way as in the HoK assertion: a `<SubjectConfirmation>` element, whose `Method` attribute is set to `holder-of-key:SSO:browser`, contains the encoded client certificate from the TLS channel. The `<SubjectConfirmation>` is inserted into the authentication request's `<Subject>` element. Due to the XML signature and the additional XML elements, the resulting HoK+ `<AuthnRequest>` messages are bigger than 2,048 bytes. Therefore, we had to change the HTTP redirect binding (i.e. transfer by HTTP GET parameter) to HTTP POST binding.

A total of 113 modified or added lines across 3 files in the SSP source code were required for these SSP modifications. Additionally, no further changes on UA were required.

7 Conclusion

Developing a secure Web SSO protocol is a nontrivial task. Due to the more complex trilateral communication flow between all participants, the attack surface is larger and more often leads to security issues compared to standard authentication. Despite the provided protection mechanisms to strengthen the authentication process, a plethora of attacks still exists.

The introduced HoK is a secure and robust Web SSO protocol that already protects against a variety of attacks (e.g. MITM). However, based on previously found authentication flaws, we showed that even the cryptographically strong security, provided by HoK, can be bypassed and thus cannot mitigate identity theft.

Our improved HoK+ approach *holistically* secures the whole SSO protocol flow and additionally mitigates RelayState Spoofing without the need of changing existing Web infrastructure (i.e. TLS, browser, and web server). Finally, the concept of HoK+ is generic and can be applied to other SSO protocols (e.g. OAuth and OpenID).

References

[ACC+08] Alessandro Armando, Roberto Carbone, Luca Compagna, Jorge Cuéllar, and M. Llanos Tobarra. Formal Analysis of SAML 2.0 Web Browser Single Sign-On: Breaking the SAML-based Single Sign-On for Google Apps. In *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering, FMSE 2008*, pages 1–10, Alexandria and VA and USA, 2008. ACM.

[ACC+11] Alessandro Armando, Roberto Carbone, Luca Compagna, Jorge Cuéllar, Giancarlo Pellegrino, and Alessandro Sorniotti. From Multiple Credentials to Browser-Based Single Sign-On: Are We More Secure? In Jan Camenisch, Simone Fischer-Hübner, Yuko Murayama, Armand Portmann, and Carlos Rieder, editors, *SEC*, volume 354 of *IFIP Advances in Information and Communication Technology*, pages 68–79. Springer, 2011.

[AWZ10] J. Altman, N. Williams, and L. Zhu. Channel Bindings for TLS. RFC 5929 (Proposed Standard), July 2010.

[Bar11] A. Barth. HTTP State Management Mechanism. RFC 6265 (Proposed Standard), April 2011.

[BBC11] Andrew Bortz, Adam Barth, and Alexei Czeskis. Origin Cookies: Session Integrity for Web Applications. In *W2SP: Web 2.0 Security and Privacy Workshop 2011*, May 2011.

[BH12] D. Balfanz and R. Hamilton. Transport Layer Security (TLS) Channel IDs. Internet-Draft, November 2012.

[BJM08] Adam Barth, Collin Jackson, and John C. Mitchell. Robust Defenses for Cross-Site Request Forgery. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, pages 75–88, New York and NY and USA, 2008. ACM.

[Can13] Scott Cantor. SAML V2.0 Channel Binding Extensions Version 1.0, 2013. `http://docs.oasis-open.org/security/`

saml/Post2.0/saml-channel-binding-ext/v1.0/cs01/samlchannel-binding-ext-v1.0-cs01.html.

[Cha06] Yuen-Yan Chan. Weakest Link Attack on Single Sign-On and Its Case in SAML V2.0 Web SSO. In *Computational Science and Its Applications - ICCSA 2006*, volume 3982 of *Lecture Notes in Computer Science*, pages 507–516. Springer Berlin Heidelberg, 2006.

[CHK+05] Scott Cantor, Frederick Hirsch, John Kemp, Rob Philpott, and Eve Maler. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, 15.03.2005, 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf.

[CKPM05] Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf, March 2005.

[DCBW12] Michael Dietz, Alexei Czeskis, Dirk Balfanz, and Dan S. Wallach. Origin-Bound Certificates: A Fresh Approach to Strong Client Authentication for the Web. In *Proceedings of the 21st USENIX conference on Security symposium*, Security'12, pages 16–16, Berkeley, CA, USA, 2012. USENIX Association.

[ERS+08] Donald Eastlake, Joseph Reagle, David Solo, Frederick Hirsch, and Thomas Roessler. XML Signature Syntax and Processing (Second Edition), 2008. http://www.w3.org/TR/xmldsig-core/.

[FGM+99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616, Hypertext Transfer Protocol – HTTP/1.1, 1999.

[GJMS08] Sebastian Gajek, Tibor Jager, Mark Manulis, and Jörg Schwenk. A Browser-based Kerberos Authentication Scheme. In Sushil Jajodia and Javier López, editors, *Computer Security - ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings*, volume 5283 of *Lecture Notes in Computer Science*, pages 115–129. Springer, August 2008.

[GP06] Thomas Groß and Birgit Pfitzmann. SAML artifact information flow revisited. Research Report RZ 3643 (99653), IBM Research, 2006. http://www.zurich.ibm.com/security/publications/2006.html.

[Gro03] T. Groß. Security analysis of the SAML Single Sign-on Browser/Artifact profile. In *Annual Computer Security Applications Conference*. IEEE Computer Society, 2003.

[Hei12] Mario Heiderich. *Towards Elimination of XSS Attacks with a Trusted and Capability Controlled DOM*. PhD thesis, Ruhr-University Bochum, Bochum, May 2012.

[KS10] Nate Klingenstein and Tom Scavo. SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0: Committee Specification 02. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-holder-of-key-browser-sso-cs-02.pdf, August 2010.

[Man03] Art Manion. Vulnerability Note VU#867593, 2003. http://www.kb.cert.org/vuls/id/867593.

[MS11] Andreas Mayer and Jörg Schwenk. Sicheres Single Sign-On mit dem SAML Holder-of-Key Web Browser SSO Profile und SimpleSAMLphp. In Bundesamt für Sicherheit in der Informationstechnik, editor, *Sicher in die digitale Welt von morgen*, pages 33–46, Gau-Algesheim, May 2011. SecuMedia Verlag.

[Pal07] Wladimir Palant. (CVE-2009-0357) XMLHttpRequest allows reading HTTPOnly cookies, 2007. https://bugzilla.mozilla.org/show_bug.cgi?id=380418.

[PW03] Birgit Pfitzmann and Michael Waidner. Analysis of Liberty Single-Sign-on with Enabled Clients. *IEEE Internet Computing*, 7(6):38–44, 2003.

[SB12] San-Tsai Sun and Konstantin Beznosov. The Devil is in the (Implementation) Details: An Empirical Analysis of OAuth SSO Systems. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 378–390, New York, NY, USA, 2012. ACM.

[Sim14] SimpleSAMLphp. SimpleSAMLphp Project. URL: http://www.simplesamlphp.org, 2007–2014.

[SMS+12] Juraj Somorovsky, Andreas Mayer, Jörg Schwenk, Marco Kampmann, and Meiko Jensen. On Breaking SAML: Be Whoever You Want to Be. In *21st USENIX Security Symposium*, Bellevue, WA, August 2012.

[The11] The National Institute of Technology. Special Publication 800-63-1: Electronic Authentication Guideline, December 2011. http://csrc.nist.gov/publications/nistpubs/800-63-1/SP-800-63-1.pdf.

[The13] The Open Web Application Security Project (OWASP). Cross-Site Request Forgery (CSRF), 2013. https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF).

[WCW12] Rui Wang, Shuo Chen, and XiaoFeng Wang. Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services. In IEEE, editor, *Security & Privacy 2012*, 2012.

Verbesserung der Netzsicherheit in virtualisierten Umgebungen mit Hilfe von OpenFlow

Andreas Brinner
genua mbH
andreas_brinner@genua.de

Rene Rietz
BTU Cottbus–Senftenberg
rrietz@informatik.tu-cottbus.de

Abstract: Viele der klassischen Techniken, mit denen die Netzsicherheit in physikalischen Systemen erhöht werden, lassen sich nicht oder nur mit großem Aufwand in virtualisierten Umgebungen einsetzen. So ist es prinzipbedingt nicht möglich, virtuelle Systeme auf einem Hostsystem physikalisch voneinander zu trennen, um deren Kommunikation durch eine Firewall filtern zu können. Auch Angriffe auf den Layern 2 und 3, wie ARP-Spoofing und Rogue-DHCP-Server, sind in physikalischen Netzen durch entsprechende Switches gut beherrschbar. In virtualisierten Umgebungen sind diese allerdings nicht einsetzbar.

In diesem Beitrag stellen wir einen Ansatz vor, mit Hilfe von OpenFlow und eines speziellen OpenFlow-Controllers die Netzsicherheit in virtuellen Systemen zu erhöhen. Ohne Veränderungen an den Gastsystemen lassen sich ARP- und DHCP-Attacken effektiv verhindern. Zum Schutz von Systemdiensten können die Datenverbindungen für die beteiligten Systeme transparent durch Firewalls, Application-Level-Gateways oder Intrusion-Detection-Systeme geroutet werden. Mit Hilfe einer Client-Authentifizierung lassen sich die definierten Sicherheitsregeln auch nach der Migration von virtuellen Instanzen weiter einhalten.

1 Einleitung

Virtualisierte Systeme sind in ihrer Standardkonfiguration oft besonders anfällig für Angriffe auf den Layern 2 oder 3. Die Linux-Bridge bietet zum Beispiel keinen Schutz gegen ARP-Attacken oder Rogue-DHCP-Server. Des Weiteren lassen sich virtuelle Gäste auf einem Host-System nicht ohne weiteres durch Firewalls voneinander trennen. Falls dies doch geschieht, dann werden die Firewalls meist ebenfalls auf dem virtuellen System ausgeführt. Dies kann aber dazu führen, dass eine Kompromittierung der Firewall den gesamten Host mitsamt aller Gastsysteme gefährdet [WDWY10].

Um die Sicherheit von virtuellen Systemen zu erhöhen, haben wir ein OpenFlow-basiertes Verfahren entwickelt, das es ermöglicht, physikalische Firewalls in die Kommunikation zwischen virtuellen Instanzen einzubinden und grundlegende Layer 2 und 3 Attacken zu unterbinden. OpenFlow [MAB+08, Ope11, Ope13b] wurde an der Stanford University entwickelt mit dem Ziel, die Möglichkeit zu erhalten, Control- und Data-Plane innerhalb

eines Switches zu trennen. Dadurch lässt sich die Switch-Logik in einen separaten Controller auslagern. Entscheidungen, beispielsweise zum Paket-Forwarding, werden nicht mehr eigenständig vom Switch, sondern vom zuständigen Controller getroffen, der dadurch die vollständige Kontrolle über das Netz und das Datenrouting erhält. Für das hier vorgestellte Verfahren wurde ein OpenFlow-Controller entwickelt, der ARP- und DHCP-Pakete an den OpenFlow-Switches nicht weiterleitet, sondern selber beantwortet. Außerdem können Regeln definiert werden, nach denen Datenverbindungen für die Clients transparent durch Firewalls umgeleitet werden können.

Unsere Lösung lässt sich zentral verwalten. Sie bedarf keiner Änderungen und keines Eingriffs auf den zu schützenden Gastsystemen. Einzig die verwendeten Switche müssen OpenFlow-fähig sein. In Linux-Hostsystemen kann dazu der Open vSwitch verwendet werden, das die Linux-Bridge ersetzt. Im Gegensatz zu einer vollständigen (virtuellen) Firewall stellt dies eine wesentlich kleinere Trusting-Computing-Base da, was Angriffe auf diese zentrale Komponente erheblich erschwert.

2 Problemstellung

Abbildung 1 zeigt ein einfaches Mustersystem, bestehend aus einem physikalischen Host und zwei virtuellen Gästen. Diese sind über ein virtuelles Netz untereinander und mit dem externen Netz verbunden. Ein Angreifer kann sich entweder im externen Netz (A) oder auf einem der Gastsysteme (B) befinden. Von extern ist ein Angriff auf das interne Netz, den Host oder eines der Gastsysteme möglich. Von einem der Gastsysteme aus, kann zusätzlich noch das externe Netz als Angriffsziel dienen. Angreifer (B) kann dabei legal Zugriff auf das Gastsystem 2 erhalten haben, zum Beispiel durch Anmieten einer virtuellen Instanz bei einem Hosting-Provider. Die Stadttore symbolisieren diejenigen Stellen, an denen idealerweise Firewalls eingesetzt werden müssten, um alle Systeme voreinander zu schützen.

Schon an diesem minimalen Mustersystem wird ersichtlich, dass für einen idealen Schutz eine große Anzahl an Firewalls benötigt wird. Folgende Fragen stellen sich und sollen als erstes geklärt werden. (1) Welchen Gefahren sind die Gastsysteme überhaupt ausgesetzt? (2) Was sind die für unsere Untersuchung relevanten Bedrohungen? Im Rahmen dieser Untersuchungen sind nur solche Bedrohungen relevant, die sich zumindest theoretisch mit Hilfe einer Firewall abwenden lassen. Dazu gehören zuallererst die klassischen Netzbedrohungen, denen auch ein physikalisches, nicht virtualisiertes System ausgesetzt ist. Danach ist (3) die Frage zu klären, ob sich für die virtuellen Umgebungen neue, systematische Angriffsmöglichkeiten ergeben, die sich mit Hilfe einer Firewall absichern lassen.

2.1 Klassische Netzbedrohungen

Im Folgenden sind typische Schwachstellen Ethernet-basierter Netze aufgeführt. Diese Schwachstellen ermöglichen es einem Angreifer vor allem, den Datenverkehr anderer Sys-

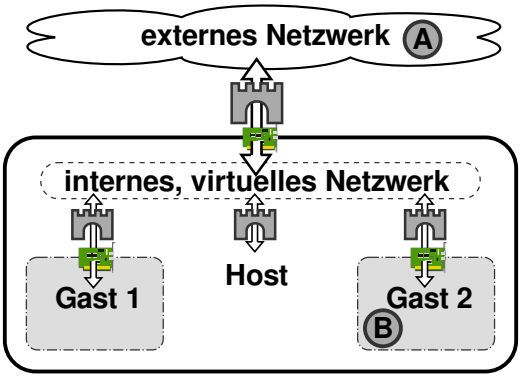


Abbildung 1: Hostsystem mit zwei virtuellen Gästen und den Positionen, an denen Firewalls zum optimalen Schutz eingesetzt werden müssten.

teme umzuleiten und mitzulesen. Sie greifen auf den Layern 2 und 3 des OSI-Modells an.

- *MAC-Spoofing.* Ein Angreifer täuscht eine falsche MAC-Adresse vor, um damit zum Beispiel per DHCP eine fremde IP-Adresse zugeteilt zu bekommen.
- *ARP-Spoofing.* Ein Angreifer versucht durch gefälschte ARP-Pakete die Kommunikation der Opfer zu unterbrechen oder derart umzuleiten, dass er sie mitlesen kann.
- *ARP-Flooding.* Durch gefälschte ARP-Pakete soll der ARP-Cache einfacher Switches geflutet werden, sodass diese in den Broadcast-Modus umschalten und alle Daten an allen Ports ausgeben. Auch hier ist wiederum das Ziel, die Kommunikation der anderen Systeme mitlesen zu können.
- *Rogue-DHCP-Server.* Der Angreifer richtet einen eigenen DHCP-Server im Netz ein und versucht auf DHCP-Requests schneller zu antworten, als der eigentliche DHCP-Server. Gelingt dies, können dem Opfer gefälschte DHCP-Pakete untergeschoben und somit dessen Konfiguration geändert werden. Durch Angabe eines gefälschten Standardgateways kann das Opfer dazu gebracht werden, den Angreifer als Standardgateway anzusehen.

Diese Schwachstellen existieren sowohl für physikalische als auch für virtualisierte Netze. Jedoch sind physikalische Systeme schon wesentlich weiter entwickelt und es existieren Lösungen, die sie besser dagegen schützen. Eigene Versuche an einem Debian-System mit KVM-Virtualisierung haben gezeigt, dass ARP-Spoofing zwischen virtuellen Gastsystemen ohne großen Aufwand möglich ist. Die standardmäßig verwendete Linux-Bridge bietet keinen Schutz gegen diese Angriffe. Ein erfolgreicher Angriff eines Rogue-DHCP-Servers basiert darauf, dass er schneller eine Antwort liefern kann, als der eigentliche DHCP-Server. Aufgrund der kurzen Wege zwischen den Gästen eines Hostsystems, ist das Einrichten eines Rogue-DHCP-Servers gerade dort sehr erfolgsversprechend.

2.2 Besondere Gefahren in virtuellen Systemen

Um herauszufinden, ob in virtualisierten Systemen neue, systematische Schwachstellen existieren, die sich durch eine Firewall absichern ließen, wurden die CVEs (Common Vulnerabilities and Exposures [MIT13]) verschiedener Virtualisierungslösungen untersucht. Dazu wurde jede Schwachstelle einem Angriffsvektor und einer Auswirkung zugeordnet. Der Angriffsvektor sagt aus, ob sich diese Schwachstelle über das Netz (”Netz”) ausnutzen lässt oder nur auf direktem Weg (”Direkt”), nachdem schon ein Zugriff auf das System besteht. Für den direkten Zugriff wird also ein Login auf dem System benötigt. Die Auswirkung unterscheidet darüber, ob ein erfolgreicher Angriff einen Zugriff auf das System ermöglicht (”Zugriff”) oder nur die Verfügbarkeit einschränkt (”Verfügbarkeit”). Je nach Anwendungsfall sind diese Auswirkungen unterschiedlich stark zu gewichten. In unserem Fall sind aber beide Auswirkungen unerwünscht und sollen verhindert werden.

	Angriffsvektoren	
Auswirkung	Netz	Direkt
Zugriff	10	41
Verfügbarkeit	8	30

Tabelle 1: Ergebnisse der CVE-Untersuchung

Tabelle 1 zeigt die Ergebnisse dieser Auswertung. Von den untersuchten 177 CVEs konnten die meisten aufgrund fehlender Informationen nicht bewertet werden. Die restlichen Schwachstellen fallen überwiegend in die Gruppe ”Direkt”, die sich nicht durch eine Firewall blocken lassen und somit für unsere Untersuchung uninteressant sind. Gerade einmal 18 Schwachstellen können über das Netz ausgenutzt werden. Dies sind unter anderem Schwachstellen im virtuellen Netztreiber, dem VNC-Zugriff auf die Gastsysteme und im DHCP-Code, die bei VMware aufgetreten sind. Die VNC- und DHCP-Schwachstellen hätten sich dabei durch eine restriktiv eingestellte Firewall schützen lassen. Die restlichen acht Schwachstellen können durch spezielle Datenpakete ausgenutzt werden und führen in sieben Fällen zu einer Denial-of-Service Attacke. Da diese Schwachstellen allerdings sehr speziell sind und sich keine Systematik dahinter erkennen lässt, dürfte es schwer bis unmöglich sein, generische Firewallregelen zu finden, die ein System proaktiv dagegen schützt.

Es bleibt also festzuhalten, dass auch virtuelle Systeme vor allem vor den klassischen Netzschwachstellen geschützt werden müssen, denen auch physikalische Systeme ausgesetzt sind. Virtuelle Systeme sind zwar vielen, neuen Angriffsmöglichkeiten ausgesetzt, es konnten allerdings keine neuen, systematischen Schwachstellen gefunden werden, die sich durch den Einsatz von Firewalls abblocken lassen. Einzig der Schutz der VNC-Konsole erscheint als sinnvoll und praktikabel.

3 Existierende Lösungsansätze

Sowohl für physikalische, als auch für virtuelle Netze existieren zumindest Teillösungen, um diese vor Angriffen zu schützen. Im Folgenden werden wir einige dieser Techniken für physikalische Netze vorstellen und auch einen Blick auf Lösungen für virtuelle Systeme werfen.

3.1 Lösungen für physikalische Netze

Um ein klassisches, physikalisches System abzusichern, werden die Netze oft in kleinere Segmente unterteilt. Dies kann zum einen auf einer logischen Ebene durch Konfiguration von IP-Subnetzen geschehen, was allerdings nur eine sehr schwache Aufteilung ist, oder zum anderen physikalisch durch entsprechende Verkabelung der Systeme. An den Übergangspunkten können dann die Daten gefiltert und untersucht werden. Dazu werden entsprechend den Sicherheitsanforderungen unterschiedliche Systeme verwendet, die allerdings auch unterschiedliche Performanceanforderungen stellen. Angefangen beim Einsatz von Routern, über einfache Paketfilter und Firewalls bis hin zu Application-Level-Gateways (ALGs) und Intrusion Detection Systemen (IDS) bieten sie einen immer besseren Schutz, indem sie immer tiefer in die zu untersuchenden Datenströme hineinschauen. Je genauer die Daten untersucht werden, desto größer wird allerdings auch der Rechenaufwand. Außerdem können alle Systeme nur solche Daten untersuchen, die durch sie hindurch gehen. Daten, die im selben Subnetz bleiben, können nicht untersucht werden.

Eine andere Möglichkeit ist der Einsatz von intelligenten Switches, die einen gewissen Schutz gegen Rogue-DHCP-Server und ARP-Attacken bieten [CIS13]. Dies wird erreicht, indem diese Switches bestimmte Datenpakete an ihren Ports verbieten. Da normalerweise jeder Netzteilnehmer an einen Switch angeschlossen ist, bietet diese Lösung eine ziemlich flächendeckende Sicherheit. Auch virtuelle LANs (VLANs) können zum Separieren von Clients eingesetzt werden, sodass nur die Systeme, die demselben VLAN zugeordnet sind, miteinander kommunizieren können. Allerdings gibt es auch für VLANs schon Angriffsmethoden, über die ein sogenanntes VLAN-Hopping erreicht werden kann [CIS13]. Außerdem ist der Einsatz von VLANs recht unflexibel, da ein Client entweder einem VLAN zugeordent ist oder nicht. Sollen mit Hilfe von VLANs alle virtuellen Gäste voneinander separiert und die gegenseitige Kommunikation gefiltert werden, so entsteht letztendlich ein sternförmiges System mit der entsprechenden Firewall im Mittelpunkt.

Soll auf den Einsatz spezieller Hardware verzichtet werden, können gewisse Konfigurationen auch statisch hinterlegt werden. Statische IP-Adressen und ARP-Tabellen können gegen die oben genannten Attacken schützen. Allerdings geht dabei ein großes Stück an Flexibilität verloren und der Administrationsaufwand steigt enorm an, da jede Änderung an allen Systemen manuell nachvollzogen werden muss.

3.2 Lösungen für virtuelle Systeme

Viele der oben genannten Lösungen lassen sich nicht ohne weiteres auf virtuelle Systeme übertragen, da sich Gäste auf einem Hostsystem nicht physikalisch voneinander trennen lassen. Auch der Einsatz von intelligenten Hardware-Switches ist dabei nicht möglich. Daher werden von den Herstellern der Virtualisierungslösungen spezielle, meist proprietäre Softwarelösungen angeboten. In [Bel99, IKBS00] wurde das Konzept der Distributed-Firewall vorgestellt. Diese besteht aus virtuellen Firewallmodulen auf den zu schützenden Systemen, die zentral administriert werden können. Die einfachere Administration wird dabei allerdings mit einem erhöhten Einrichtungsaufwand erkauft, da auf jedem zu schützenden System diese Firewallmodule installiert werden müssen. Außerdem können nur solche Systeme geschützt werden, für die ein passendes Firewallmodul existiert.

4 Lösung mit Hilfe von OpenFlow

Mit Hilfe von OpenFlow wurde eine Lösung für virtuelle Systeme entwickelt, die diese gegen die zuvor genannten Angriffsformen absichert. Sie erfordert keine Änderungen an den Gastsystemen, ist flexibel, zentral administrierbar, verhindert effektiv Angriffe auf den Layern 2 und 3 und ermöglicht es, externe physikalische Firewalls in die virtuellen Systeme einzubinden. Dazu wird in einem ersten Schritt die Linux-Bridge durch den Open vSwitch [OPE13a] ersetzt (siehe Abbildung 2). Dies ist ein OpenFlow-fähiger virtueller Switch, der Switchingfunktionalität in virtuellen Systemen zur Verfügung stellt. Werden auch die physikalischen Switches durch OpenFlow-fähige Modelle ersetzt, entsteht ein wesentlich homogeneres Netz, in dem die Grenzen zwischen physikalischer und virtueller Umgebung verschwimmen. Abhängig von definierbaren Sicherheitsregeln können einzelne oder alle Datenverbindungen eines Systems für dieses transparent durch eine Firewall geroutet werden. Dafür kann sowohl eine virtuelle als auch eine physikalische Firewall eingesetzt werden.

Die Switches haben vollständige Kontrolle über alle Datenströme im Netz und werden ihrerseits vom zentralen OpenFlow-Controller gesteuert. Dieser ist somit die zentrale und kontrollierende Instanz, die außerdem eine globale Sicht auf das Netz hat. Einige Sicherheitsfunktionen können auch direkt von den Switchen bzw. dem Controller übernommen werden. Im Folgenden werden die weiteren Ideen und Maßnahmen vorgestellt, auf denen das Prinzip des IP-Switches beruht.

4.1 ARP- und DHCP-Pakete nicht mehr broadcasten

Eines der großen Probleme in virtuellen Netzen besteht darin, dass ARP- und DHCP-Pakete im Netz gebroadcastet werden und somit jeder Netzteilnehmer mithören und auch antworten kann. Es ist also naheliegend, in den OpenFlow-Switches entsprechende Regeln zu hinterlegen, die dafür sorgen, dass alle ARP- und DHCP-Pakete an den Control-

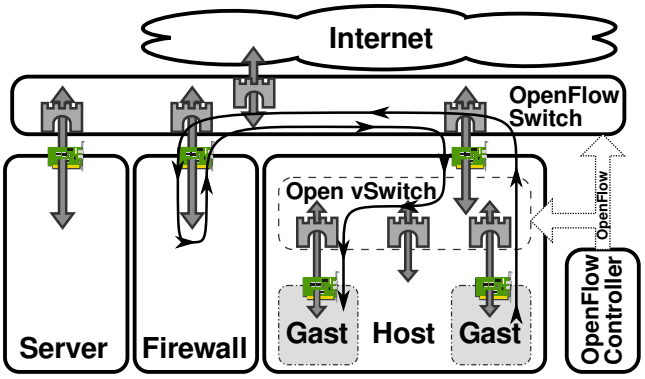


Abbildung 2: Virtueller und physikalischer Switch mit OpenFlow Controller. Die Kommunikation zwischen den beiden virtuellen Instanzen wird über die physikalische Firewall umgeleitet.

ler umgeleitet und nicht mehr weiter verteilt werden. Um berechtigte Anfragen dennoch beantworten zu können, wird im Controller ein entsprechender ARP- und DHCP-Server integriert. Ein Großteil der Schwachstellen lässt sich alleine mit dieser einfachen Maßnahmen beheben. ARP-Spoofing wird unterbunden, indem (bösartige) APR-Pakete nicht länger im Netz gebroadcastet, sondern vom Controller selbst beantwortet oder verworfen werden. Das gleiche gilt für DHCP-Pakete, was Rogue-DHCP-Server verhindert. Ein Angreifer sieht nicht einmal mehr die DHCP-Requests der anderen Systeme, was notwendig wäre, um im richtigen Augenblick eine bösartige Antwort senden zu können.

4.2 Verwenden von IP- anstelle von MAC-Adressen

Eine weitere Idee ist, zum Ansprechen der Zielsysteme anstelle von MAC-Adressen IP-Addressen zu verwenden. Viele der genannten Schwachstellen beruhen darauf, dass zur Adressierung eines Systems zwei unterschiedliche Ebenen zuständig sind (Layer 2 und 3), die miteinander in Einklang gebracht werden müssen. Um zum Beispiel die zu einer IP-Adresse zugehörige MAC-Adresse zu erhalten, wird ARP verwendet. Genau an diesem Punkt kann ein Angreifer ansetzen, um das System in einen nicht konsistenten Zustand zu bringen und Verbindungen mitzulesen oder zu unterbinden.

Dazu wird der ARP-Server im Controller so implementiert, dass er alle ARP-Anfragen der Clients mit einer virtuellen, nicht existierenden MAC-Adresse (im Beispiel FF:...) beantwortet. Beim Versand von Ethernet-Frames verwenden die Clients diese MAC-Adressen zusammen mit den IP-Adressen der Zielrechner. Die OpenFlow-Switches leiten diese Informationen an den Controller weiter, der anhand der IP-Adresse eine passende Route bestimmt und entsprechende Match-Einträge in den Switches erstellt. Der letzte Switch auf dieser Route erhält die Aufgabe, die MAC-Adressen so umzuschreiben, dass sie für das

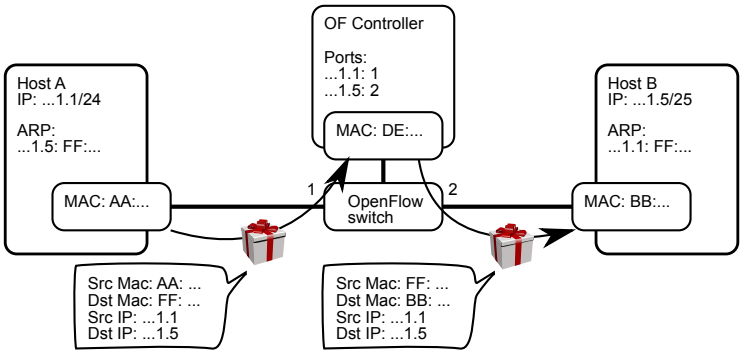


Abbildung 3: Kommunikation mit virtuellen MAC-Adressen und OF-Switch.

Zielsystem gültig sind. Dieser Vorgang ist beispielhaft in Abbildung 3 dargestellt. Aufgrund der virtuellen MAC-Adressen kennt jeder Client nur seine eigene, reale Adresse. Von den anderen Systemen ist ihnen nur deren IP-Adresse bekannt und auch die eigentliche Netztopologie bleibt ihnen verborgen. Jedes System kennt somit nur all jene Informationen, die für eine erfolgreiche Kommunikation zwingend erforderlich sind.

4.3 Einbinden von Firewalls

Einfache Paketfilterregeln können direkt durch entsprechende Regeln auf den OpenFlow-Switches implementiert werden. Auch Firewalls mit zustandsbehafteter Verbindungsverwaltung lassen sich mit Hilfe des Controllers und der Regeln auf den Switches realisieren. Im Prinzip entspricht ein Flow-basiertes Switching einer solchen Firewall. Für jede neue Datenverbindung wird eine Anfrage an den Controller gesendet, der dann im Einzelfall entscheiden kann, ob diese Verbindung zugelassen werden soll oder nicht. Danach wird eine spezifische Regel auf den Switches hinterlegt. Sollen Application-Level-Gateways oder Intrusion Detection Systeme eingesetzt werden, so ermöglicht es dieselbe Technik, die zu filternden Verbindungen dynamisch und transparent zu einem entsprechenden System zu routen. Erst nach erfolgreicher Filterung werden die Daten dann dem Zielsystem zugestellt. All diese Sicherheitsregeln lassen sich im Controller für jeden Client individuell hinterlegen und werden unabhängig davon, an welchem Port sich das System mit dem Netz verbindet, durchgesetzt. Jeder Port an jedem OpenFlow Switch wird somit zur Firewall (siehe auch Abbildung 2).

4.4 MAC-Adressen der Clients authentifizieren

Um die Clients eindeutig identifizieren zu können, kann der Controller diese mit Hilfe von 802.1x oder durch Überprüfung des SSH-Hostkeys authentifizieren. Dies geschieht nachdem sich ein Client am Netz angemeldet hat. Mit Hilfe der Client-Authentifizierung kann MAC-Spoofing verhindert werden, was wiederum notwendig ist, wenn im Controller für jeden Client individuelle Sicherheitsregeln hinterlegt werden sollen. Dadurch wird ein freies Roaming oder Migrieren von Client-Systemen ermöglicht, wobei alle Sicherheitsregeln mitwandern und beibehalten werden.

5 Umsetzung und Evaluation

Im Rahmen des Projekts ist mit Hilfe des Ryu-Frameworks [RYU13] ein OpenFlow-Controller entstanden, der diese Funktionalität umsetzt. Während der Entwicklungsphase wurde der Controller intensiv mit Mininet [LHM10] getestet. Später kam ein eigenständiges Testnetz mit Debian/KVM-Server, mehreren virtuellen Instanzen und einer Anbindung an zwei OpenWRT Linksys Router hinzu. Dieses System ist in Abbildung 4 schematisch dargestellt. Auf dem KVM-Server sind drei virtuelle Gastsysteme mit unterschiedlichen Aufgaben eingerichtet. Das NAT-Gateway ist über das Interface eth1 mit dem externen Netz verbunden und verbindet die Teilnehmer im OpenFlow-Netz mit diesem. In der Firewall-Instanz ist eine genuscreen-Firewall installiert, die für die Clients transparent den ein- und ausgehenden Datenverkehr filtert. Der virtuelle Client dient als Testinstanz eines normalen Netzteilnehmers. Eine zweite genuscreen-Firewall wurde als externes, physikalisches System angeschlossen.

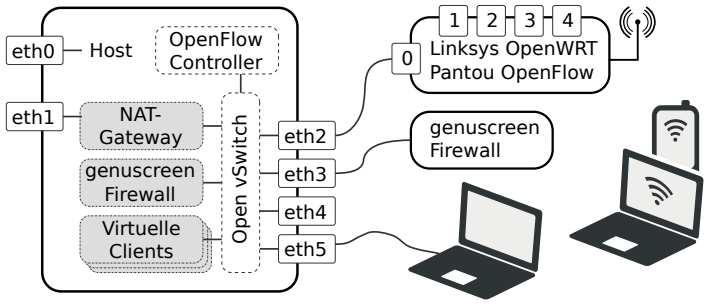


Abbildung 4: OpenFlow Testsystem bestehend aus Debian/KVM Host mit virtuellen Instanzen, Linksys Router, physikalischer Firewall und Endgeräten.

Über den Open vSwitch sind neben diesen Instanzen noch vier weitere, physikalische Netzschnittstellen angebunden, die somit als OpenFlow-Switch arbeiten. An diese Ports können direkt weitere Endgeräte angeschlossen werden, die sich dann wie normale Teilnehmer des OpenFlow-Testnetzes verhalten. Außerdem können weitere OpenFlow-Swit-

ches kaskadiert werden. In Abbildung 4 ist ein Linksys-Router mit einer Pantou OpenWRT Firmware eingezeichnet, der sich wiederum als OpenFlow-Switch verhält und das Testnetz auch für WiFi-Teilnehmer zur Verfügung stellt. Darüber konnten weitere Clients, wie Laptops oder Smartphones, erfolgreich ins OpenFlow-Testnetz eingebunden und vom OpenFlow-Controller gesteuert werden. Das Routing der Datenverbindungen durch die Firewall und/oder das NAT-Gateway wird durch den Controller gesteuert und geschieht für die Clients völlig transparent.

Durch die in 4.1 beschriebenen Maßnahmen konnten Angriffe wie ARP-Spoofing, ARP-Flooding und Rogue-DHCP-Server effektiv verhindert werden. Die im OpenFlow-Controller implementierten ARP- und DHCP-Server verwerfen ungültige APR- und DHCP-Pakete, sodass andere Netzteilnehmer diese nicht mehr erhalten. Angriffe auf Systemdienste konnten durch das Einbinden einer Firewall in die Kommunikationsverbindung verhindert werden. Die Daten wurden dabei für die Netzteilnehmer transparent durch eine der Firewalls geroutet und gefiltert. MAC-Spoofing konnte durch Authentifizierung der Netzteilnehmer mit 802.1X verhindert werden. Dazu wurde der OpenFlow-Controller entsprechend erweitert, sodass er Systeme, die sich neu am Netz anmelden, authentifizieren konnte.

6 Zusammenfassung und Ausblick

Es konnte ein OpenFlow-Controller entwickelt werden, der erfolgreich die aufgezeigten Fähigkeiten umsetzt. Die Funktionalität wurde in unterschiedlichen Test-Setups, unter anderem im Mischbetrieb mit virtuellen und physikalischen Systemen, getestet. Da für den Testbetrieb keine dedizierte OpenFlow-Hardware, sondern Softwareswitches verwendet wurden, kann keine realistische Performanceabschätzung für den Mischbetrieb gegeben werden. Für den reinen virtuellen Betrieb ergab der Einsatz des Open vSwitches nur geringe Performanceeinbußen. Der Einfluss des OpenFlow Controllers auf die Performance wird erst nach einer Optimierung des Quellcodes realistische Ergebnisse liefern können.

In weiteren Untersuchungen soll der Mischbetrieb mit OpenFlow- und Legacy-Switches untersucht werden. Außerdem soll die weitere Einbindung von Middleboxen, wie der Firewall, weiter untersucht und entsprechende APIs definiert werden. Dadurch könnten unter anderem Hardwareshunts auf den Switches realisiert werden, die bei einer erkannten, gutartigen Verbindung die Performance enorm steigern können.

Literatur

[Bel99] Steven M. Bellovin. Distributed Firewalls. *login*, Seiten 37–39, November 1999.

[CIS13] CISCO. VLAN Security White Paper. `http://www.cisco.com/en/US/products/hw/switches/ps708/products\_white\_%paper09186a008013159f.shtml`, 2013.

[IKBS00] Sotiris Ioannidis, Angelos D. Keromytis, Steve M. Bellovin und Jonathan M. Smith. Implementing a Distributed Firewall. *Proceedings of Computer and Communications Security (CCS)*, Seiten 190–199, November 2000.

[LHM10] Bob Lantz, Brandon Heller und Nick McKeown. A Network in a Laptop: Rapid Prototyping for Software-defined Networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, Seiten 19:1–19:6, New York, NY, USA, 2010. ACM.

[MAB+08] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker und Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, Marz 2008.

[MIT13] MITRE Corporation. CVE – Common Vulnerabilities and Exposures. `http://cve.mitre.org/`, 2013.

[Ope11] OpenFlow Consortium. OpenFlow – Enabling Innovation in Your Network. `http://archive.openflow.org/`, 2011.

[OPE13a] Open vSwitch – An Open Virtual Switch. `http://openvswitch.org/`, 2013.

[Ope13b] Open Networking Foundation. OpenFlow. `http://www.opennetworking.org/sdn-resources/onf-specifications/openflow%`, 2013.

[RYU13] Ryu SDN Framework. `http://osrg.github.io/ryu/`, 2013.

[WDWY10] Hanqian Wu, Yi Ding, C. Winer und Li Yao. Network security for virtual machine in cloud computing. In *Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on*, Seiten 18–21, 2010.

Deploying Static Application Security Testing on a Large Scale

Achim D. Brucker and Uwe Sodan
{achim.brucker, uwe.sodan}@sap.com

SAP AG
Central Code Analysis Team
Dietmar-Hopp-Allee 16
D-69190 Walldorf

Abstract: Static Code Analysis (SCA), if used for finding vulnerabilities also called Static Application Security Testing (SAST), is an important technique for detecting software vulnerabilities already at an early stage in the software development life-cycle. As such, SCA is adopted by an increasing number of software vendors.

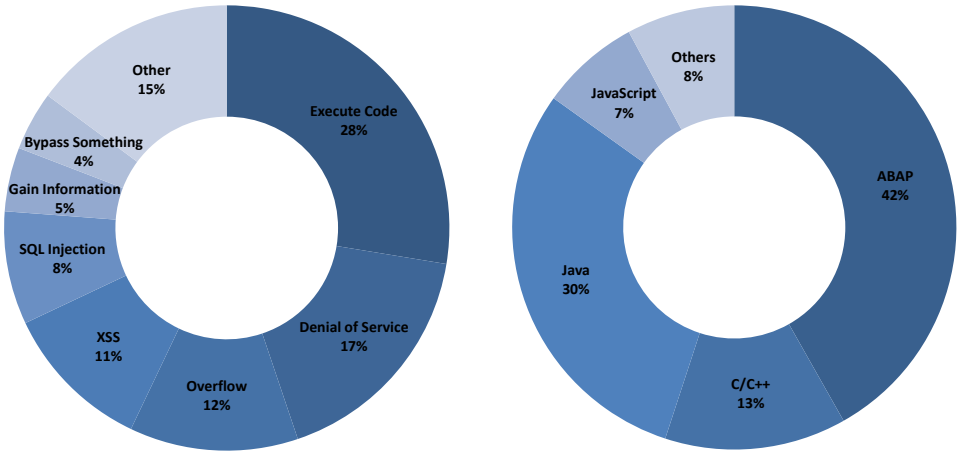
The wide-spread introduction of SCA at a large software vendor, such as SAP, creates both technical as well as non-technical challenges. Technical challenges include high false positive and false negative rates. Examples of non-technical challenges are the insufficient security awareness among the developers and managers or the integration of SCA into a software development life-cycle that facilitates agile development. Moreover, software is not developed following a greenfield approach: SAP's security standards need to be passed to suppliers and partners in the same manner as SAP's customers begin to pass their security standards to SAP.

In this paper, we briefly present how the SAP's Central Code Analysis Team introduced SCA at SAP and discuss open problems in using SCA both inside SAP as well as across the complete software production line, i. e., including suppliers and partners.

1 Introduction

Software security is becoming an increasingly important differentiator for IT companies. Consequently, techniques for preventing software vulnerabilities during system development are becoming more and more important. Figure 1a shows the distribution of vulnerabilities since 1999 according to [Nat]. The majority of the vulnerabilities (e. g., most types of code execution, overflows, XSS, SQL injection), are caused by programming errors and, thus, counter-measures need to be taken on the implementation level.

Static Code Analysis (SCA), if used for finding vulnerabilities also called *Static Application Security Testing* (SAST), is a technique that analyses programs without actually executing them (i. e., in contrast to dynamic testing approaches) and, thus, can be applied in early stages of the development life-cycle where fixing of vulnerabilities is comparatively cheap [GK02]. Currently, SAP focuses on applying SCA for finding potential security vulnerabilities as early as possible. In principle, an SCA tool analyses the source code of a



(a) Vulnerability types of CVE reports since 1999

(b) Analysed source languages at SAP

Figure 1: Overview of reported vulnerabilities and analysed source languages

software component (e. g., an application, library) and reports potential security problems: the set of findings. For each finding, an human expert is necessary to decide:

- If the finding represents a vulnerability, i. e., a weakness that can be exploited by an attacker (*true positive*), and, thus, needs to be fixed.
- If the finding cannot be exploited by an attacker (*false positive*) and, thus, does not need to be fixed.

Similarly, if an SCA tool does not report security issues, this can have two reasons:

- The source code is secure (*true negative*)
- The source code has security vulnerability but due to limitations of the tool, the tool does not report a problem (*false negative*).

In an ideal setting, the number of false positives and false negatives should be zero, i. e., the tool should be *sound* (i. e., no false positives) and *complete* (i. e., no false negatives). In practice, this cannot be achieved both due to fundamental reasons (e. g., undecidability of the halting problem) as well as compromises that need to be made between (potential) conflicting goals such as performance and correctness. Besides these, rather technical challenges, there are also non-technical challenges that make the introduction of SCA techniques difficult in a large development organisation with more than 12 000 developers.

In this paper, we report on our experiences in introducing SCA at Europe's largest software vendor: SAP. In more detail, we motivate the choice for SCA based on SAP's product security standard (Sec. 2) as well as present SAP's code scan strategy (Sec. 3). Thereafter, we discuss our approach in introducing and integrating SCA into SAP's software development life-cycle (Sec. 4) and discuss problems that we are facing (Sec. 5). Finally, we summarise our lessons learned and draw conclusions (Sec. 6).

```
<%  DATA: ffs TYPE tihttpnvp, ff  TYPE  ihttpnvp.
    DATA: lv_name type IHTTPNAM ,        " input parameter
          lv_value type IHTTPVAL .       " input parameter
    request → get_form_fields( changing fields = ffs ).
    DELETE ffs WHERE name cs '~'.
    LOOP AT ffs INTO ff.
      lv_name = cl_abap_dyn_prg ⇒ escape_xss_xml_html( ff-name ).
      lv_val  = cl_abap_dyn_prg ⇒ escape_xss_xml_html( ff-val ). %>
      doc.writeln('<input␣type="hidden"␣name="<%=ff-name␣%>"␣value="<%=ff-val␣%>">');
<%    CLEAR lv_name.
      CLEAR lv_val.
    ENDLOOP. %>
```

Listing 1: An ABAP Business Server Page that contains XSS vulnerabilities. The input variables `ff-name` and `ff-val` are directly written into the DOM and, thus, an attacker that can influence these inputs can, e. g., inject arbitrary JavaScript into the web browser of the victim. Note the incorrect use of the sanitation function `escape_xss_xml_html` which does not sanitise the content of the variables `ff-name` and `ff-val`.

2 SAP's Product Standard "Security"

The overall goal of the software development life-cycle of SAP is to deliver secure, reliable, and high-performance software that is easy to use and fulfils the demands of SAP's customers. The non-product specific requirements such as usability, security, or performance are described in different *Product Standards*. The goal of SAP's *Product Standard "Security"* is to ensure that SAP products are secure by default as well as easy to operate securely. A prerequisite to achieve this goal is to ensure that SAP products are free of implementation related security vulnerabilities such as SQL injection, path traversal, memory corruption, or buffer overflow. Listing 1 shows an ABAP Business Server Page that contains a XSS vulnerability: user-influenced input from the input `ff-name` is written into the DOM without any checks. We refer the reader to the OWASP Top Ten [OWA13] or the CVE/SANS top 25 [Mit11], for a more comprehensive overview of vulnerability types.

Besides secure programming guidelines [SAP13b] and security training courses for developers and product owners, the use of Static Application Security Testing (SAST) (or, more general, Static Code Analysis (SCA)) is one of the key techniques that contributes to achieving this goal. Overall, SAST helps to detect implementation related security vulnerabilities in an early phase of the software development. In general, for finding implementation related security vulnerabilities, there are mainly two types of approaches (for a general introduction to SAST and competing approaches, we refer the reader elsewhere, e. g., [PCFY07, CW07]): 1. dynamic testing that analyses an application while the application is executed and 2. static code analysis that analyses the application without executing it. After an analysis of these techniques, SAP decided to introduce SAST as mandatory step for preventing implementation related security vulnerabilities. At SAP is mandatory (with the exception of technical reasons such as the use of technologies that are not supported by the available SAST tools) for products developed at SAP. The main reasons for this decisions are that SAST can be automated to a large degree, scales for large application (i. e., a large code basis), yields in repeatable results, does not require isolated test systems of

whole applications, and allows for fixing problems early in the development life-cycle. This decision is also backed up by our own studies in comparing, e. g., SAST and pen testing approaches as well as empirical studies done outside of SAP, e. g., [SWJ13]. Of course, SAST is only one building block of a holistic security testing approach [BB14].

3 SAP's Code Scan Strategy

The use of SAST at SAP is organised and governed by the *Central Code Analysis Team* that is responsible the code scan strategy. The code scan strategy is guided by the following four principles:

1. Security code scans are *embedded into the development process*, thus, should be as less disruptive for the developers as possible.
2. It is understood that there is always a compromise between accuracy and efficiency of the tools.
3. A manual review of the findings is necessary which requires human effort. The false positives are marked permanently so that manual review has to be done only once.
4. The tools will prioritise the findings and the development teams will concentrate on analysing and fixing all high priority findings; low priority findings will be spot checked, moved to the backlog and fixed as allowed by the capacities.

From these principles, the other responsibilities of the Central Code Analysis Team are derived. The two most important ones are:

- monitor the tool market, evaluate and select the tools that are used at SAP. If necessary, drive the development of own tools if no suitable tools can be identified.
- constantly monitor the efficient and effective use of the tools with the goal of developing and implementing improvements. These improvements include both the processes and the tools itself (including their configuration).

Thus, from an organisational perspective, this team acts as the first contact for security related code analysis (dynamic and static) for all stakeholders of SAP. From a technical perspective, the Central Code Analysis Team defines (and constantly improves) the default configurations of the tools, evaluates and introduces new tools or new versions of the tools.

If SAST cannot be applied to a product (e. g., it is written in programming language not supported by the a SAST tool), the development team has to develop a mitigation plan, i. e., describe the alternative techniques they apply to ensure the security of the product.

4 Introducing Static Application Security Testing at SAP

In this section, we briefly introduce the software development life-cycle at SAP and explain the processes that ensure an efficient and effective use of SAST by the development teams.

4.1 Integrating SAST into the Software Development life-cycle of SAP

The software development at SAP follows an agile and decentralised approach, i. e., the development teams are empowered to organise the details of their development process as long as they achieve requirements of the various products standards of SAP. The compliance to these standards is reviewed at globally defined milestones, called *Quality Gates*. Moreover, the quality gates allow to synchronise the various sub-process of the software development life-cycle at SAP, called *Idea-to-Market* (I2M) (see Figure 2):

1. The *Idea-to-Portfolio* (I2P) process describes how ideas from different sources inside and outside SAP are collected, matured and turned into a solution portfolio.
2. The *Portfolio-to-Solution* (P2S) process describes the path from given portfolio decision to the customer available product. This includes the definition of the product scope, release of the budget, as well as the actual development of the product. This process contains three sub-processes: *Planning and Setup*, *Development*, and *Release Preparation* and the quality gates are:
   (a) *Portfolio-to-Planning* (P2P): Among others, this quality gate includes an approval of the actual business case.
   (b) *Planning-to-Development* (P2D): Handover from the planning teams to the development teams which, among others include the commitment that the necessary development resources are available.
   (c) *Development-to-Production* (D2P): This quality gate defines the end of the development of a specific product version. After passing this quality gate, the software is either prepared for on premise delivery or prepared for hosting. Here, the product groups need to prove at this point in time that they comply with the Code Scan Strategy, i. e., scanned and analysed their product and, more importantly, fixed the identified (and prioritised) vulnerabilities.
   (d) *Production-to-Rollout* (P2R): After passing this quality gate, the solution is finished to be rolled-out to customers. Before the solution is rolled-out, the *Quality, Governance, and Production Team* validates, again, that the solution complies with the various products standards at SAP. For security, e. g., this includes a penetration test of the finished product.
3. The *Solution-to-Market* (S2M) process describes the necessary steps to bring a developed solution to the market.

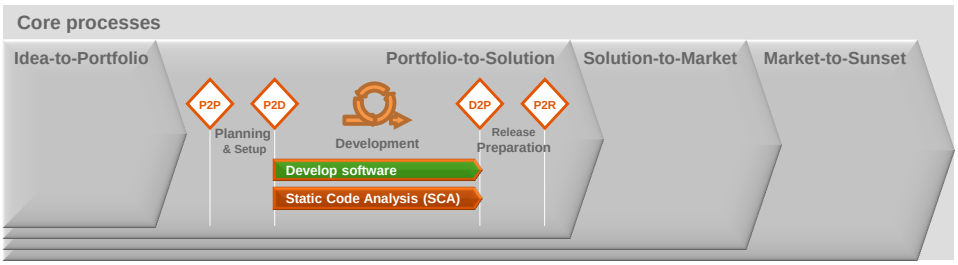


Figure 2: The core of the software development life-cycle at SAP

4. The *Market-to-Sunset* (M2S) process defines the controlled end-of-life of a solution as well as the maintenance of the solution.

As SAST helps to prevent vulnerabilities caused by bad programming, i. e., on the implementation level, we concentrate our SAST effort on the Portfolio-to-Solution (P2S) phase. As we will see in the next section, customer feedback received in the Market-to-Sunset (M2S) phase plays an important role for improving and focusing the use of SAST at SAP.

4.2 Continuous Improvement of our SAST Approach

Introducing SAST is not a one-time effort: only the constant monitoring and improvement of both the tools as well as the processes can ensure the *efficiency* and *effectiveness* of SAST as technique for preventing vulnerabilities. The main goals are increasing

- the technological scope, i. e., developing support for new programming languages or development frameworks.
- the security scope, i. e., by developing checks for new vulnerability types.
- the efficiency by lowering the false positive and false negative rates. Similarly, the prioritisation of the findings needs to be monitored and improved.
- the usability of the tools, e. g., develop better integration into the develop environments ore improving the response time of central servers supporting scans or audits.

Measuring the efficiency and effectiveness of SAST is difficult, defining and measuring good key performance indicators (KPIs) is even more difficult. Besides recording the basic data (number of issues found, time required for fixing issues, etc.), we concentrate our efforts for improving our SAST approach on direct collaborations with various stakeholders:

- the *Product Security Response Team* which is the main contact point for customers and external security researchers reporting security issues of SAP products and, thus, manages the process for analysing the root cause of all externally reported vulnerabilities and ensures that they are fixed in a timely manner.
- the *Quality, Governance, and Production Team* which validates (e. g., using pen tests) before shipment that SAP's products comply to the product standard "Security."
- the *Development Teams*, including their security experts, that execute the processes defined by the Code Analysis Team, i. e., they actually use the various SAST tools.

Besides the constant improvements of the usability, together with the development teams, we are concentrating on improving the efficiency of the tools by

- *Reducing the number of false positives:* False positives are detected by the development teams (together with their security experts) during the audit and might, e. g., caused by new security APIs not yet recognised by the scan tool. Whenever they recognise changes in the number of false positives (or have other suggestions for improvements) they should contact the centrla code scan team and collaborate in improving the current tool configuration (or search for better alternatives).
- *Reducing the number of false negatives:* False negatives are much harder to detect, as the tool does not report them. To improve in this area, we concentrate on vulner-

Table 1: SAST Tools used at SAP

Programming Language	Tool	Vendor
ABAP	CVA (SLIN_SEC) [SAP13a]	SAP
C/C++	Coverity [Cov13]	Coverity
All other languages	Fortify [HP13]	HP

abilities reported by external researchers as well as vulnerabilities found internally by alternative security testing approaches (e. g., the validation done by our Quality, Governance, and Production Team).
For each found vulnerability, we check if it could, potentially, be detected by a SAST tool. If it could be found, we analyse (together with the stakeholders) why it was not detected during the regular SAST efforts and how we can improve the situation. The outcome of this analysis can be that the vulnerability

- is not detected by the most current configuration. In this case, we check (together with the tool vendor) how the configuration or tool can be improved to report this issues in the future.
- was reported by the default cans but not fixed prior to shipment. In this case, we investigates the reason for not fixing the issue. This might result in an updated version of the tool configurations (e. g., re-prioritising certain issues types).
- The vulnerability was reported for a product release that was developed before the code scan strategy was implemented. In this case, we check if the vulnerability is detected by the current implementation of the code scan strategy.

Moreover, for products that were not scanned during development, the number of reported vulnerabilities that can be detected by SAST is an additional argument for justifying the efforts of implementing the Code Scan Strategy.

The effort in improving the tooling is prioritised according the size of the code base. ABAP is still the most used language at SAP (see Figure 1b). These priorities can change rather quickly: a few years ago, JavaScript was only used rarely. During the last two years, the use of JavaScript, both client-side and server-side, increased significantly and JavaScript is, in the near future, about to replace C/C++ as the third most used language at SAP.

4.3 Where We Are Today

The implementation of SAP's code scan strategy requires over 2 000 developers and security experts to work, on a regular basis (scan and audit), with SAST tools—even more developers are involved in fixing vulnerabilities found.

SAP uses three different SAST tools (see Table 1) across all development units and the whole product range (ranging from mobile apps to on-premise offering to on-demand offerings). Until today, more than two billion lines of code were scanned and the findings analysed and, if exploitable, fixed. To customers, the most visible result of implementing a rigorous

code scan strategy was the Security Patch Day in December 2010 were SAP published over 500 patches (called *SAP Security Note*) [SAP10]. This, of course, requires a careful planning and agreement with customers. In the meantime, patches fixing vulnerabilities detected by SAP internal code scans that are released using regular support packs. This reduces the effort that customer require to apply these patches significantly.

For Coverity and Fortify, the central code scan team operates a company-wide server infrastructure for executing and archiving code audits. Close to 3 000 software projects (each of them having multiple versions) are stored on these central servers. For ABAP the scans are deeply integrated into the development tools (e. g., the ABAP Test Cockpit) and, thus, no additional central infrastructure is necessary.

To ensure the security across the whole software supply chain, SAP makes also use of external services (e. g., offered by Veracode [Ver13]) for analysing third-party code that is not available as source code. Moreover, we regularly evaluates alternatives to the tools used (e. g., Checkmarx [Che13] or IBM Security AppScan Source [IBM13]), e. g., to increase the number of supported programming languages.

5 Open Issues

Low security awareness and hard to estimate risk of not fixing security issues. Even though SAP has rolled out mandatory security training courses for all developers since 2011, the security awareness during regular development is insufficient. Moreover, it is often hard, if not impossible, to estimate the potential financial risk of not fixing a security issues. Thus, particular, in cases where development efforts need to be prioritised between the investigation security issues and implementing new (functional) features, teams try to minimise the effort spend for analysing or fixing security issues.

Pushing SAST across the software supply chain. To increase the software security of SAP's offerings, we cannot concentrate our efforts on SAP's software development: we need to include suppliers and partners to ensure the security of

- third party software (including commercial offerings, Freeware, and Free and Open-Source Software) used in the development,
- partner products sold by SAP or used by SAP customers together with SAP solutions.

Finally, our customers start to impose similar requirements to us. Thus, we needs to be able to provide a proof of our security initiatives, e. g., the successful execution of our code scan strategy.

Huge and hybrid multi-language applications. Many modern applications are built by composing many different components that are not necessarily written in the same language. For example, a Software-as-a-Service (SaaS) offering might be based on a user-interface written in JavaScript and a backend that uses both a Java and ABAP stack. Thus, large applications statically does not only raise the question of how to modularise (i. e., al-

lowing for scanning each component in isolation while still allowing for a precise analysis of dataflows across several components) scan. It does as well raise the question of how to analyse dataflows across different programming and framework contexts.

Dynamic programming paradigms and languages. Modern applications are highly distributed and often based on asynchronous and dynamic programming concepts and languages, e. g., JavaScript. These large (millions lines of code) JavaScript applications can neither be handled satisfactory with state-of-the-art static nor with dynamic approaches.

Lack of standardised regression test suites. New versions of SAST tools (as well as alternatives tools) can have a significantly different behaviour (e. g., not reporting issues that were reported previously, reporting a large number of false positives). Thus, we requires an effective and efficient way for evaluating (testing) security testing tools. This includes artificial test suites such as the SAMATE test suites [SAM13] (only available for Java and C/C++) as well as real products that use different programming styles. Building and maintaining such test suites is non-trivial and labour-intensive. Similarly, test methods for the rigorous testing of technique for SAST configurations (e. g., cleansing rules) as well as security functions (e. g., cleansing or sanitation functions) needs to be developed.

6 Conclusion and Lessons Learned

While Bessey et al. [BBC+10] present the perspective of a vendor (i. e., Coverity) of a static code analysis tool "a few billions lines of code later," we discussed the users perspective based on analysing several billions lines of code.

In our opinion, applying SAST is a never-ending journey requiring constant monitoring and improvements. Thus, while there are still many open questions (recall Sec. 5), we believe that, based on our experiences, we can already provide valuable insights and recommendations on how to introduce and improve SAST in an agile software development life-cycle. To begin with, we recommend to follow the five keys to success identified by Chandra et al, that we briefly (see [CCS06] for details) recite here:

1. *Start small.* Work closely with one group to sort out any kinks in the adoption process, and once this pilot group succeeds, move on to replicate that success.
2. *Go for the throat.* Rather than trying to stomp out every security problem on the first day, pick a handful of things that go wrong most often and go after them first.
3. *Appoint a champion.* Find a developer who knows a little bit about every part of the system and likes to mentor. Put your new champion in charge of the tool adoption.
4. *Measure the outcome.* Tools spit out quantifiable results, so use them to track projects over time. Code analysis results can indicate where trouble is brewing, but it can also tell you whether your education and training efforts are paying off.
5. *Make it your own.* Use the tool to enforce your own standards and guidelines. Spend some time investigating customisation capabilities to achieve deeper inspection and maximised accuracy for your environment.

In our experience, these five keys are only the first step and we suggest to at another seven and, thus, to complete the dozen:

6. *Plan and invest enough resources.* Introducing SAST requires significant resources, not only for analysing and fixing findings. You need also resources for educating developers, operating tool infrastructure. Finally, you might need to introduce new expert roles (e. g., security experts) to your organisation.
7. *Plan and invest enough infrastructure.* Providing a reliable, fast, and scalable solution for executing the scans as well as storing and managing the results requires a significant amount of hardware. Depending on the tool (e. g., if local audit and scans are supported), not only powerful servers might be required, also an upgrade of the development machines of the developers might necessary.
8. *Do understand your developers as you allies.* While some processes suggested by SAST vendors suggest that developers and security experts are fighting against each other, we strongly recommend to understand team as one team fighting for a secure product: SAST should be a tool that supports your developers in producing high-quality software and, thus, should be introduced in collaboration with them.
9. *Execute scans regularly.* SAST is not a one-time effort that solves all security issues on-and-for-all. The efficient and effective use of SAST requires its deep integration into the development process and tools. Code scans should be executed and monitored regularly—transferring audit results from one scan to the next works best for small changes of code basis. Thus, apply code scans nightly or at least weekly.
10. *Plan your changes and updates.* Do not introduce changes in the scan infrastructure (new versions of tools or updated to their configuration) within one development cycle. The change in results that might be caused by this may reduce the confidence of the developers into the tool. Always plan for such changes ahead and introduce them together with your developers.
11. *Do get support (and commitment) from your management.* Introducing SAST is a significant investment that requires long-term support from management (recall the memo from Bill Gates [Gat02]). While investment into security often needs to be balanced with other investments, it must be clearly understood by all stakeholders (including management and developers) that SAST can only improve the product security if and only if it is understood as a continuous long-term investment.
12. *Do not stop here.* Introducing SAST can only be the first step. SAST needs to be complemented with other techniques (e. g., threat modelling, pen testing) and, thus, needs to be embedded into a *holistic security testing strategy* [BB14] which, in itself, should be part of a secure software development life-cycle.

This dozen keys to success need to be implemented by clearly defined processes and supported by the responsible roles. Watch out, these keys are necessary but not sufficient for the successful introduction and execution of SAST in a large software development unit.

Overall, static code scans are useful and contribute to avoid vulnerabilities that, otherwise, would be found be external researchers and, thus, could harm the reputation of SAP.

References

[BB14] Ruediger Bachmann and Achim D. Brucker. Developing Secure Software: An Holistic Approach to Security Testing. *Datenschutz und Datensicherheit*, March 2014.

[BBC+10] Al Bessey, Ken Block, Ben Chelf, Andy Chou, Bryan Fulton, Seth Hallem, Charles Henri-Gros, Asya Kamsky, Scott McPeak, and Dawson Engler. A few billion lines of code later: using static analysis to find bugs in the real world. *Commun. ACM*, 53:66–75, February 2010.

[CCS06] P. Chandra, B. Chess, and J. Steven. Putting the tools to work: how to succeed with source code analysis. *Security Privacy, IEEE*, 4(3):80–83, may-june 2006.

[Che13] Checkmarx. Checkmarx. `http://www.checkmarx.com/`, 2013. Site visited on 2013-12-07.

[Cov13] Coverity. Coverity. `http://www.coverity.com/`, 2013. Site visited on 2013-12-07.

[CW07] Brian Chess and Jacob West. *Secure programming with static analysis*. Addison-Wesley Professional, first edition, 2007.

[Gat02] Bill Gates. Memo from Bill Gates. `http://www.microsoft.com/en-us/news/features/2012/jan12/gatesmemo.aspx`, January 2002.

[GK02] M.P. Gallaher and B.M. Kropp. The Economic Impacts of Inadequate Infrastructure for Software Testing. Technical Report Planning Report 02-03, National Institute of Standards & Technology, May 2002.

[HP13] HP. HP/Fortify. `http://www.hp.com/us/en/software-solutions/software-security/index.html`, 2013. Site visited on 2013-12-07.

[IBM13] IBM. IBM Security AppScan Source. `http://www-03.ibm.com/software/products/en/appscan-source/`, 2013. Site visited on 2013-12-07.

[Mit11] Mitre. 2011 CWE/SANS Top 25 Most Dangerous Software Errors. `http://cwe.mitre.org/top25/`, 2011. Site visited on 2013-12-02.

[Nat] National Institute of Standards and Technology (NIST). National Vulnerability Database. `http://nvd.nist.gov/`. Site visited on 2013-11-24.

[OWA13] OWASP. OWASP Top Ten. `https://www.owasp.org/index.php/Top_10_2013-Top_10`, 2013. Site visited on 2013-12-02.

[PCFY07] M. Pistoia, S. Chandra, S. J. Fink, and E. Yahav. A survey of static analysis methods for identifying security vulnerabilities in software systems. *IBM Systems Journal*, 46(2):265–288, 2007.

[SAM13] SAMATE. SAMATE Reference Data Set. `http://samate.nist.gov/SRD/testsuite.php`, 2013. Site visited on 2013-12-07.

[SAP10] SAP. SAP Note 1533030: Patch Day 12/2010: General Info for SAP Suite and NetWeaver, 2010.

[SAP13a] SAP AG. SAP NetWeaver Application Server, add-on for code vulnerability analysis. `http://scn.sap.com/docs/DOC-48613`, 2013. Site visited on 2013-11-24.

[SAP13b] SAP AG. SAP Secure Programming Guides. `http://scn.sap.com/docs/DOC-48612`, 2013. Site visited on 2013-11-24.

[SWJ13] Riccardo Scandariato, James Walden, and Wouter Joosen. Static analysis versus penetration testing: a controlled experiment. In *Proceedings of the 24th IEEE International Symposium on Software Reliability Engineering,*, pages 1–10. IEEE, November 2013.

[Ver13] Veracode. Veracode Independent Software Audits. `http://www.veracode.com/services/application-audits.html`, 2013. Site visited on 2013-12-07.

Einsatz von digitaler Forensik in Unternehmen und Organisationen

Stefan Meier, Günther Pernul

Lehrstuhl Wirtschaftsinformatik I - Informationssysteme
Universität Regensburg
Universitätsstraße 31
D-93053 Regensburg
stefan.meier@ur.de
guenther.pernul@ur.de

Abstract:
Zielgerichtete und hoch spezialisierte Angriffe auf die IT-Systeme erschweren es zunehmend die Systeme abzusichern und für alle Angriffsszenarien entsprechende Maßnahmen im Vorfeld zu etablieren. Aus diesem Grund ist es nötig die Täterverfolgung zu forcieren, anstatt nur in abwehrende IT-Sicherheitsmaßnahmen zu investieren. Zur Verfolgung von Tätern eignen sich besonders Methoden aus dem Bereich der digitalen Forensik. Mithilfe der durch digitale forensische Untersuchungen gewonnenen hieb- und stichfesten digitalen Beweise ist es nicht nur möglich einen IT-Sicherheitsvorfall umfassend aufzuklären, sondern auch die Täterverfolgung einzuleiten. Die gesammelten digitalen Spuren können dann letzten Endes auch vor Gericht verwendet werden. Inwieweit Organisationen bereits in der Lage sind solche digitalen forensischen Untersuchungen durchzuführen ist bisher in der Literatur jedoch noch nicht umfassend geklärt. Deshalb wurde im Rahmen der in diesem Paper präsentierten Studie der tatsächliche Status Quo der digitalen Forensik in Unternehmen und Organisationen umfassend untersucht. Die Ergebnisse der Studie zeigen, dass aktuell noch ein erheblicher Aufholbedarf besteht. Weiter hat die Studie gezeigt, dass auch die Wissenschaft gefordert ist, entsprechende Methoden bereitzustellen.

1 Motivation und Zielsetzung

Zielgerichtete Angriffe auf Unternehmensnetze wie der Angriff auf das Playstation Network[1] von Sony oder der erfolgreiche Angriff auf den IT-Sicherheitsspezialisten RSA[2] (EMC Corporation) haben gezeigt, dass es für Unternehmen unabdingbar ist sich mit dem Thema digitale Forensik zu beschäftigen. Die erfolgreichen Angriffe zeigen, dass heute kein Unternehmensnetzwerk vor einem Angriff sicher ist. Auch die Abschottung von IT-Systemen vom Internet ist keine ausreichende Garantie mehr für die Sicherheit von IT-

[1] Update on PlayStation Network and Qriocity `http://blog.us.playstation.com/2011/04/26/update-on-playstation-network-and-qriocity/` (Zugriff am 30.01.2014)

[2] Frequently asked questions about rsa securid, `http://www.emc.com/collateral/guide/11455-customer-faq.pdf` (Zugriff am 30.01.2014)

Systemen. Dies hat z.B. der Stuxnet-Wurm[3] eindrucksvoll gezeigt. Der bei einem solchen zielgerichteten Angriff entstehende Schaden kann dabei enorme und für manche Organisationen auch existenzbedrohende Ausmaße annehmen [BSJ10]. Dabei ist der direkt zu beziffernde Schaden oft nur die Spitze des Eisberges [ABB+12]. Neben rechtlichen Konsequenzen wie Schadensersatzforderungen oder Strafen kann der Ruf einer Organisation enorm leiden. Um schnell und zielgerichtet auf einen erfolgreichen Angriff reagieren zu können, bieten sich Methoden aus der digitalen Forensik an. Neben einer umfassenden Aufklärung eines IT-Sicherheitsvorfalls können im Rahmen von digitalen forensischen Untersuchungen auch Spuren bzw. Beweise zur Täterverfolgung gerichtsfest gesichert werden. In vielen Publikationen wird die Verwendung von digitaler Forensik zur Aufklärung von IT-Sicherheitsvorfällen in Organisationen propagiert und gilt teilweise auch bereits als gegeben [Row04, PIP10]. In anderen Arbeiten wird dagegen von einem erheblichen Nachholbedarf von Unternehmen gesprochen [MGL11]. Der tatsächliche Stand der Integration und Verwendung von digitaler Forensik in Organisationen wurde jedoch noch nicht umfassend untersucht. Ziel der in diesem Paper präsentierten Studie ist es deshalb den tatsächlichen Stand der Integration und Verwendung von digitaler Forensik in Organisationen zu klären. Weiter sollten auch Hinweise auf Forschungslücken im Bereich der digitalen Forensik in Organisationen gefunden werden.

Das Paper ist wie folgt aufgebaut: Im nächsten Abschnitt 2 werden verwandte Arbeiten bzw. Themengebiete aufgezeigt. In Abschnitt 3 werden das Design sowie die Ziele der Studie im Detail vorgestellt. Die Ergebnisse der Studie werden dann in Abschnitt 4 vorgestellt und in Abschnitt 5 diskutiert. Im letzten Abschnitt 6 folgt dann eine kurze Zusammenfassung des Papers und ein Ausblick auf weitere Themen im Gebiet der Unternehmensforensik.

2 Verwandte Arbeiten

Mit der Integration und Verwendung von digitaler Forensik in Organisationen beschäftigt sich insbesondere das Gebiet der *Forensic Readiness*. Die Ziele der *Forensic Readiness* sind zum einen den Nutzen der gesammelten digitalen Spuren zu erhöhen und zugleich die Kosten für Untersuchungen zu verringern [Tan01]. Um *Forensic Readiness* zu erreichen, wurden in der Literatur bereits verschiedene Maßnahmen vorgeschlagen. Ein umfassendes Werk mit Vorschlägen zur Implementierung von *Forensic Readiness* ist [Row04]. Rowlingson legt dabei in einer eher pragmatischen Sichtweise die benötigten Richtlinien und Vorgehensweisen für Unternehmen in seinen zehn Schritten zur *Forensic Readiness* dar. In [Row04] wird zudem gezeigt, dass digitale Forensik eine Ergänzung und Verbesserung zu bestehenden Aktivitäten wie dem Risikomanagement, dem Vorfallsmanagement, der IT-Sicherheitsüberwachung sowie der Aus- und Weiterbildung sein kann. Außerdem wird nach einem Vorfall ein explizites Lernen der Organisation von einem Vorfall gefordert [Row04]. Eine weitere Arbeit aus dem Bereich *Forensic Readiness* ist [YM01]. Yasinsac und Manzano schlagen Richtlinien zur Verbesserung digitaler foren-

[3] The Real Story of Stuxnet, `http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet` (Zugriff am 30.01.2014)

sischer Untersuchungen nach dem Eintreten von Vorfällen der Computerkriminalität vor. Weiter wird eine bessere Täterverfolgung gefordert, die nach Ansicht der Autoren durch Computer- und Netzwerkforensik ermöglicht wird [YM01]. Daneben ist auch ein Bewusstsein für IT-Sicherheitsrichtlinien, sowie insbesondere deren Einhaltung im Management nötig [WWW03]. Die Einbettung von digitaler Forensik in das IT-Sicherheitsmanagement und insbesondere das Vorfallsmanagement ist ein weiterer nötiger Aspekt [WWW03, PMB03, PK10]. Zudem muss verhindert werden, dass Geschäftsabläufe massiv gestört und unterbrochen werden [PMB03]. Um dies sicherzustellen muss digitale Forensik im und vom Unternehmen selbst betrieben werden und darf nicht komplett an Dritte ausgelagert werden [PMB03]. Mit dem Fokus auf kleine und mittlere Unternehmen stellen Barske et al. Maßnahmen zur Implementierung von *Forensic Readiness* vor [BSJ10]. Dabei bedienen sich Barske et al. größtenteils der Maßnahmen und Konzepte aus bereits vorhandenen Werken im Bereich *Forensic Readiness*, wie z.B. [Tan01] oder [Row04] und zeigen diese für den Kontext der kleinen und mittleren Unternehmen auf. Aufgrund der vielen unterschiedlichen Initiativen, sowohl in Organisationen wie auch bei öffentlichen Einrichtungen wird in [MGL11] eine Standardisierung innerhalb des Bereichs *Forensic Readiness* gefordert.

Um die Reife eines Unternehmens hinsichtlich der *Forensic Readiness* festzustellen, wird von Chryssanthou und Katos ein Reifegradmodell vorgestellt [CK12]. Je nachdem welche Maßnahmen zur Vorbereitung einer digitalen forensischen Untersuchung im Unternehmen implementiert sind, wird das Unternehmen auf einer Skala von 0 bis 5 eingeordnet. 0 bedeutet, dass nahezu keine Vorbereitungen getroffen wurden und 5 bedeutet, dass über alle notwendigen Maßnahmen hinaus Vorbereitungen getroffen wurden.

Für die konkrete Durchführung von digitalen forensischen Untersuchungen wird neben den vorbereitenden Maßnahmen vor allem ein Vorgehensmodell benötigt. In [Cas11, S. 187ff], [FS07] und [DF11, S. 57ff] finden sich verschiedene Vorgehensmodelle zur Durchführung von Untersuchungen. Grundsätzlich sind die Phasen Vorbereitung, Identifizierung, Konservierung, Untersuchung und Analyse sowie Präsentation in fast allen Modellen, jedoch teilweise in unterschiedlichen Ausprägungen enthalten [Cas11, S. 189f]. Eine allgemein akzeptierte und standardisierte Vorgehensweise gibt es jedoch bisher nicht, obwohl insbesondere von Dewald und Freiling auf die besondere Wichtigkeit eines Vorgehensmodells hingewiesen wird [DF11, S. 84].

Zusammenfassend bieten die vielen Veröffentlichungen ein breites Spektrum an Maßnahmen welche in Organisationen implementiert werden könnten. In manchen Publikationen wird deshalb auch bereits von der digitalen Forensik als zentrales Element der ITK-Infrastruktur gesprochen [PIP10]. Die tatsächliche Integration von Maßnahmen der digitalen Forensik in Organisationen wurde jedoch noch nicht genauer untersucht.

3 Studiendesign und Durchführung

Wie bereits in den vorangegangenen Abschnitten erwähnt soll diese Studie klären, ob sich Organisationen tatsächlich bereits aktiv mit dem Thema digitale Forensik beschäftigen.

Weiter sollte geklärt werden, ob und für welchen Zweck digitale Forensik in Organisationen verwendet wird, wenn es bereits eingesetzt wird.

Da die digitale Forensik in Organisationen oft mit IT-Sicherheitsvorfällen in Verbindung gebracht wird sollten auch Beziehungen zum Bereich IT-Sicherheit überprüft werden. Insbesondere die im Rahmen einer Zertifizierung des IT-Sicherheitsmanagements nach ISO 27001 oder BSI Grundschutz umgesetzten Maßnahmen gelten als Basisvoraussetzungen für digitale forensische Untersuchungen [Nel06], weshalb hier auch die Details etwaiger Beziehungen genauer mit untersucht werden sollten.

Sollte eine Organisation bisher noch keine forensischen Untersuchungen durchgeführt haben bzw. entsprechende Methoden zur *Forensic Readiness* noch nicht implementiert haben sollte geklärt werden, welche Gründe gegen den Einsatz von digitaler Forensik sprechen bzw. welche Probleme es bei der Implementierung vorbereitender Maßnahmen gibt.

Zusammenfassend lassen sich folgende Forschungsfragen ableiten, die mithilfe der Studie geklärt werden sollten:

1. Haben sich Organisationen bereits mit der Thematik digitale Forensik beschäftigt und entsprechende Maßnahmen und Ressourcen etabliert?
2. Zu welchem Zweck werden Methoden der digitalen Forensik verwendet?
3. Sehen die Organisationen Beziehungen zum IT-Sicherheitsmanagement?
4. Welche Gründe stehen dem Einsatz von digitaler Forensik in Organisationen entgegen?

Zur Durchführung der Studie wurde ein Fragebogen mit insgesamt 21 Fragen erstellt. Der Fragebogen war sowohl als „Offlineversion“ in Papierform verfügbar als auch online mithilfe des Systems SoSci Survey[4] modelliert. Potentielle Teilnehmer wurden entweder per E-Mail eingeladen den Onlinefragebogen auszufüllen oder auf Veranstaltungen mit dem Themenschwerpunkt IT-Sicherheit direkt akquiriert. Insgesamt konnten im Untersuchungszeitraum vom 15.05.2013 bis 24.06.2013 **69 Organisationen** erfolgreich befragt werden.

4 Ergebnisse

Da die Studie im deutschsprachigen Raum durchgeführt wurde ist es nicht überraschend, dass von den 69 befragten Organisationen 65 ihren Hauptsitz in Deutschland haben. Von den verbliebenen vier Organisationen hat je eine Organisation ihren Sitz in Frankreich, Japan, der Türkei und in den Vereinigten Staaten. Abbildung 1 zeigt die Größe der Unternehmen gemessen an deren Mitarbeiterzahl und der zugehörigen Branche. Deutlich zu sehen ist in Abbildung 1, dass es gelungen ist Organisationen aus verschiedenen Branchen und mit unterschiedlicher Größe zu akquirieren. Die Unternehmensgröße ist in Form der

[4] https://www.soscisurvey.de/

Mitarbeiterzahl angegeben, da manche Unternehmen die Frage nach dem Umsatz der Organisation nicht beantwortet haben. Im Folgenden und bei allen Hypothesentests wird deshalb als Kennzahl für die Unternehmensgröße immer die Anzahl der Mitarbeiter verwendet. Bei allen Hypothesentests gilt zudem eine Irrtumswahrscheinlichkeit von $\alpha = 0,05$. Das Hypothesenpaar ist dabei definiert als H_0: Die Merkmale sind unabhängig. H_1: Die Merkmale sind abhängig.

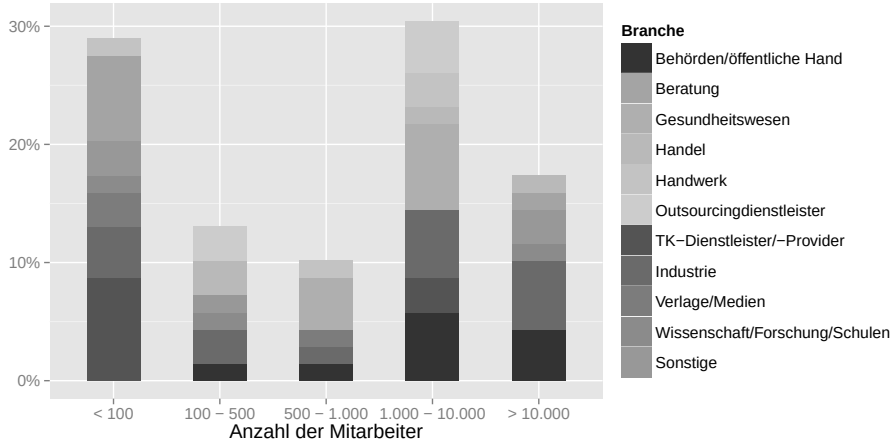


Abbildung 1: Teilnehmer nach Anzahl der Mitarbeiter und Branche

Neben der Mitarbeiterzahl wurde das jährliche Budget für IT-Sicherheit abgefragt. Dabei sind die Ausgaben für IT-Sicherheit höher, je größer ein Unternehmen ist (Spearmans Rangkorrelationskoeffizient (r_s) = 0,82). Dieser Zusammenhang ist auch statistisch signifikant bei einem p-Wert von 0,00. Zwischen der Branche und den Ausgaben für IT-Sicherheit beträgt der Kontingenzkoeffizient (K) 0,77, jedoch muss aufgrund eines p-Wertes von 0,20 ein statistischer Zusammenhang abgelehnt werden.

Die zentrale Frage nach der Implementierung bzw. Verwendung von digitaler Forensik sollte zum einen über die Frage nach der bewussten Implementierung von Maßnahmen aus der *Forensic Readiness* beantwortet werden und außerdem über die Frage, ob im Unternehmen bereits digitale forensische Untersuchungen durchgeführt wurden bzw. dies zukünftig geplant ist. Abbildung 2 zeigt die Antworten auf die beiden Fragen in einer kombinierten Darstellung. Die Balken zeigen jeweils den Prozentsatz der Antworten auf die Frage nach der Implementierung von *Forensic Readiness*. Die Antwort auf die Frage, ob im Unternehmen bereits eine digitale forensische Untersuchung durchgeführt wurde wird über die Füllung der Balken in Abbildung 2 dargestellt. Abhängig von der Antwort auf die erste Frage wurde die Antwort auf die zweite Frage dem entsprechenden Balken zugeordnet. Lediglich 15,93 % der Organisationen haben demnach bereits Maßnahmen aus dem Bereich *Forensic Readiness* implementiert. 40,60 % der Organisationen planen zukünftig konkrete Maßnahmen zu implementieren und in 43,47 % der Organisationen spielt das Thema *Forensic Readiness* überhaupt keine Rolle. Digitale forensische Untersuchungen haben bereits 37,68 % der befragten Organisationen durchgeführt. In 62,32 % der

Organisationen wurden noch keine Untersuchungen durchgeführt und es sind auch keine digitalen forensischen Untersuchungen geplant.

Betrachtet man Abbildung 2 genauer, so lässt sich eine Korrelation zwischen der Durchführung von Untersuchungen und der Implementierung bzw. Planung von Maßnahmen der *Forensic Readiness* vermuten. Die Korrelation lässt sich mit einem r_s von 0,58 bestätigen und ist statistisch signifikant bei einem p-Wert von 0,00. Eine besondere Abhängigkeit zwischen einer Zertifizierung nach ISO 27001 bzw. BSI Grundschutz und der Durchführung digitaler forensischer Untersuchungen kann dagegen nicht nachgewiesen werden.

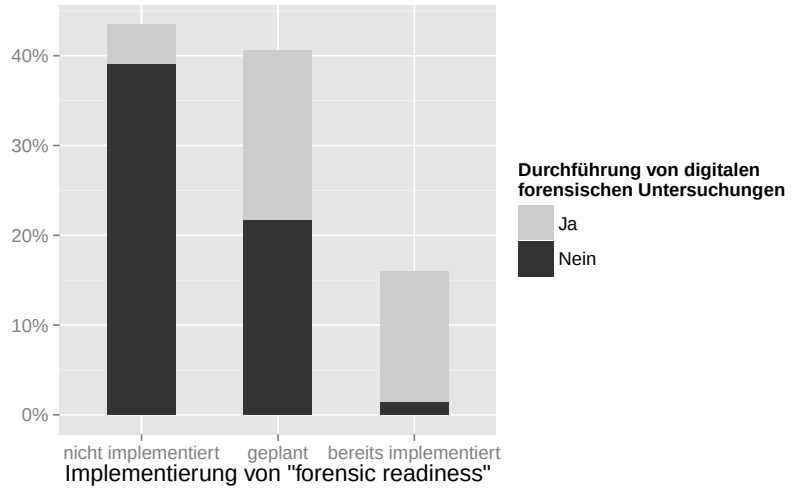
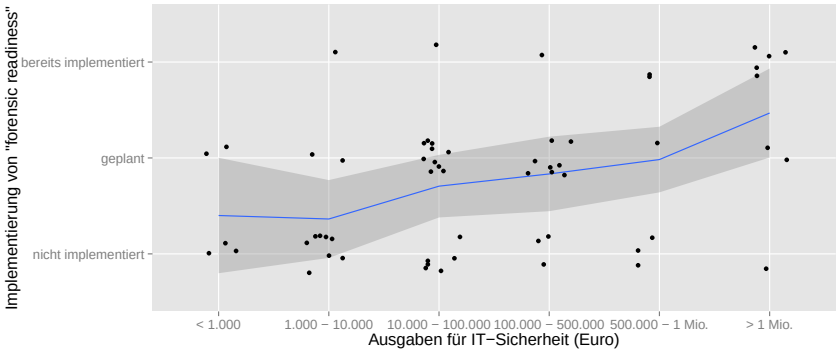


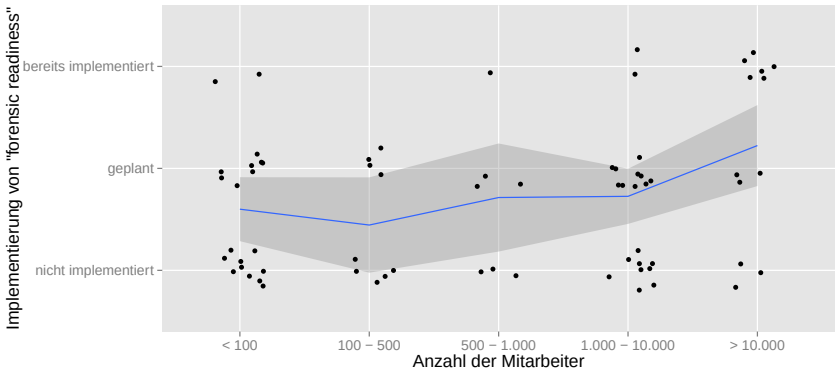
Abbildung 2: Implementierung von *Forensic Readiness* und Durchführung digitaler forensischer Untersuchungen

Hinsichtlich der Implementierung von *Forensic Readiness* kann weiter eine Korrelation zur Größe einer Organisation als auch zu den IT-Sicherheitsausgaben nachgewiesen werden. Abbildung 3(a) zeigt die Ausgaben für IT-Sicherheit in Relation zur Implementierung von *Forensic Readiness*. Die Korrelation ist bei r_s = 0,41 (p-Wert = 0,00) eher schwach. Dasselbe gilt für die Abhängig der *Forensic Readiness* von der Unternehmensgröße, dargestellt in Abbildung 3(b). Hier kann lediglich eine sehr schwache aber statistisch signifikante Korrelation von r_s = 0,24 (p-Wert = 0,02) nachgewiesen werden. Eine Abhängigkeit des Implementierungsgrades von der Branche muss jedoch abgelehnt werden (K = 0,63, p-Wert = 0,20). Weiter hat auch die Bedeutung von IT-Systemen für den Geschäftsbetrieb nur eine geringe (r_s = 0,30, p-Wert: 0,01) Auswirkung auf die Implementierung von Maßnahmen der *Forensic Readiness*. Eine Zertifizierung nach ISO 27001 bzw. BSI Grundschutz und die Implementierung bzw. geplante Implementierung von Maßnahmen der *Forensic Readiness* korrelieren dagegen mit einem r_s von 0,48 (p-Wert = 0,00).

Neben den präventiven Maßnahmen der IT-Sicherheit sollte auch geprüft werden, ob die befragten Organisationen tatsächlich bereits Opfer von Computerkriminalität bzw. digitalen Angriffen wurden. Eine Organisation hat bei dieser Frage keine Auskunft erteilt. Von



(a) *Forensic Readiness* und Ausgaben für IT-Sicherheit



(b) *Forensic Readiness* und Unternehmensgröße

Abbildung 3: Ausgaben für IT-Sicherheit bzw. Unternehmensgröße in Relation zur *Forensic Readiness*

den verbleibenden 68 Organisationen wurden 40,0 % bereits Opfer von Angriffen. Wenn eine Organisation bereits einem Angriff auf ihre IT-Systeme zum Opfer gefallen ist, so könnte man meinen, dass diese sich auf zukünftige Angriffe besser vorbereitet. Ein Zusammenhang zwischen der Implementierung von *Forensic Readiness* und der Opfer von Angriffen kann jedoch nicht nachgewiesen werden (r_s = 0,13, p-Wert = 0,15). Auch die Ausgaben für die IT-Sicherheit sind bei Organisationen die bereits angegriffen wurden nicht zwingend höher als bei den anderen 60 % der noch nicht wissentlich erfolgreich angegriffenen Organisationen (r_s = 0,24, p-Wert = 0,07). Die Gruppe der noch nicht erfolgreich angegriffenen Organisationen wurde zudem befragt, ob die Organisation in naher Zukunft mit einem Angriff rechnet. Dabei gaben 44,0 % der Organisationen an, dass sie nicht mit einem Angriff rechnen und 56,0 % der Organisationen rechnen mit einem Angriff. Die Organisationen, die mit einem Angriff rechnen bereiten sich jedoch nicht zwingend besser auf digitale forensische Untersuchungen vor als die Organisationen die nicht mit einem Angriff rechnen (r_s = 0,19, p-Wert = 0,12).

4.1 Einsatz von digitaler Forensik in Organisationen

Abbildung 4 zeigt im Detail zu welchen Zwecken Methoden der digitalen Forensik in Organisationen verwendet werden. Am häufigsten verwenden Organisationen demnach digitale Forensik tatsächlich zur Täteridentifikation nach einem IT-Sicherheitsvorfall gefolgt von einem expliziten Lernen von einem digitalen Angriff zur Verbesserung der IT-Sicherheit. Darauf folgend wird digitale Forensik zur Untersuchung von Verstößen gegen interne Richtlinien verwendet und danach zur Aufklärung von klassischen Verbrechen. Bei der Häufigkeit von digitalen forensischen Untersuchungen gibt es keinen Zusammenhang zwischen großen Organisationen und häufigeren Untersuchungen ($r_s = 0{,}19$, p-Wert = 0,23). Weiter haben Organisationen die regelmäßig forensische Untersuchungen durchführen nicht zwingend interne personelle Ressourcen um diese Untersuchungen durchzuführen ($r_s = 0{,}32$, p-Wert = 0,20).

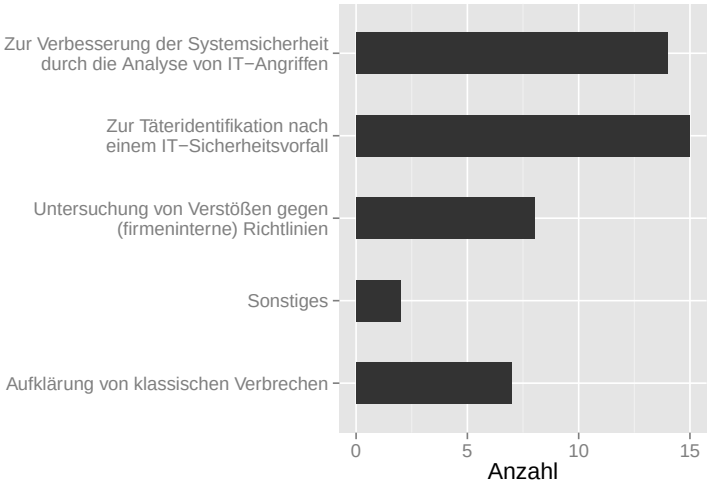


Abbildung 4: Verwendung von digitaler Forensik in Organisationen

Die Höhe der Ausgaben für IT-Sicherheit und die Höhe der Ausgaben für digitale Forensik korrelieren dagegen sehr stark bei einem $r_s = 0{,}87$ und einem p-Wert von 0,00.

4.2 Gründe für das Fehlen von digitalen forensischen Maßnahmen in Organisationen

Neben den im vorigen Abschnitt 4.1 beschriebenen konkreten Zielen von digitalen forensischen Untersuchungen wurde im Rahmen der Studie auch explizit nachgefragt wieso digitale Forensik gerade nicht eingesetzt wird. Abbildung 5 zeigt die Antworten auf diese Frage. Demnach wird digitale Forensik insbesondere aufgrund fehlender personeller und/oder finanzieller Mittel nicht eingesetzt gefolgt von fehlenden Best Practices für digi-

tale forensische Untersuchungen. In 15 Organisationen ist das Thema nicht einmal bekannt und vier Organisationen halten digitale Forensik sogar für nicht geeignet.

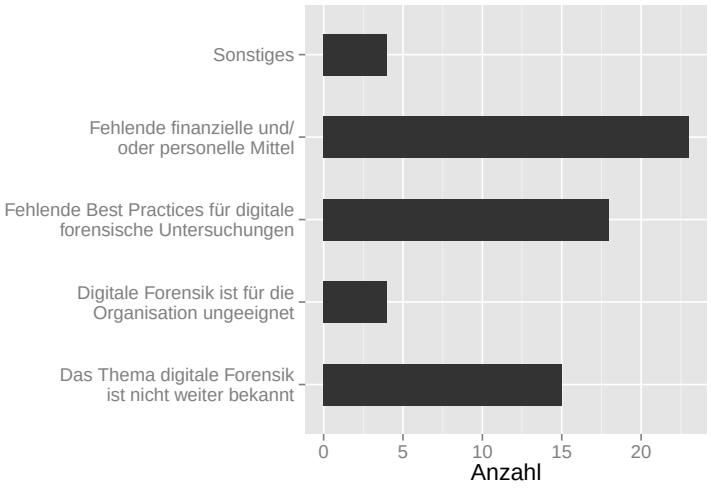


Abbildung 5: Gründe wieso digitale Forensik in Organisationen nicht zum Einsatz kommt

5 Bewertung der Ergebnisse

Im Folgenden sollen nun die in Abschnitt 3 aufgeworfenen Forschungsfragen mithilfe der Ergebnisse aus dem vorigen Abschnitt 4 diskutiert und beantwortet werden.

Frage 1: Haben sich Organisationen bereits mit der Thematik digitale Forensik beschäftigt und entsprechende Maßnahmen und Ressourcen etabliert?
Die Antwort auf diese Frage liefern die Ergebnisse aus der direkten Frage nach der Implementierung von Maßnahmen aus dem Bereich *Forensic Readiness* sowie der Nachfrage hinsichtlich der Durchführung von digitalen forensischen Untersuchungen. Die Antworten zeigen, dass manche, insbesondere große Organisationen sich bereits mit dem Thema auseinander gesetzt haben. Weiter wurden in 37,68 % der befragten Organisationen bereits digitale forensische Untersuchungen durchgeführt. Zusammenfassend lässt sich sagen, dass sich manche Organisationen bereits mit dem Thema digitale Forensik beschäftigt haben. In der Mehrzahl der Organisationen ist jedoch keine Forensikkompetenz vorhanden und das Thema digitale Forensik ist eher unwichtig.

Frage 2: Zu welchem Zweck werden Methoden der digitalen Forensik verwendet?
Forschungsfrage 2 lässt sich erschöpfend über Abbildung 4 beantworten. Es zeigt sich, dass digitale Forensik tatsächlich für die Täteridentifikation verwendet wird. Aber auch

die Kontrolle und Durchsetzung von internen Richtlinien mittels Methoden der digitalen Forensik zeigt, dass das breite Anwendungsspektrum zumindest in einigen wenigen Organisationen bereits ausgeschöpft wird.

Frage 3: Sehen die Organisationen Beziehungen zum IT-Sicherheitsmanagement?
Verbindungen zwischen der IT-Sicherheit und der digitalen Forensik können über das zur Verfügung stehende Budget für das jeweilige Thema nachgewiesen werden. So ist eine positive Korrelation zwischen der Höhe der Ausgaben für IT-Sicherheit und dem Budget für digitale Forensik nachweisbar. Was die Vorbereitung auf digitale forensische Untersuchungen mittels Methoden aus der *Forensic Readiness* angeht kann zumindest eine schwache positive Beziehung zu den Ausgaben für IT-Sicherheit festgestellt werden. Auch gibt es eine sehr schwache Korrelation zwischen einer vorhandenen Zertifizierung nach BSI Grundschutz bzw. ISO 27001 und der Implementierung von *Forensic Readiness*. Zusammenfassend können durchweg eher schwache bis mittelstarke Beziehungen zwischen den Gebieten nachgewiesen werden.

Frage 4: Welche Gründe stehen dem Einsatz von digitaler Forensik in Organisationen entgegen?
Neben den heute schon genutzten Möglichkeiten der digitalen Forensik hat die Studie insbesondere gezeigt wieso das Thema digitale Forensik gerade keine Rolle in der Mehrzahl der befragten Organisationen spielt. Abbildung 5 beantwortet diese Frage direkt. Neben fehlenden personellen wie auch finanziellen Mitteln sind insbesondere fehlende Best Practices Gründe wieso die digitale Forensik eine solch geringe Durchdringung erfährt. Auch ist es eher beunruhigend, dass manche Organisationen die digitale Forensik für ungeeignet halten bzw. das Thema noch gar nicht in der Organisation bekannt ist.

6 Fazit und Ausblick

Die Studie hat gezeigt, dass digitale Forensik in manchen Organisationen bereits in der vollen Breite genutzt wird. Viel wichtiger für die Wissenschaft ist jedoch die Erkenntnis, dass insbesondere Best Practices fehlen und auch die notwendigen personellen wie auch finanziellen Mittel zur Einführung und Nutzung von digitaler Forensik anscheinend noch zu hoch sind. Auch ist in vielen Organisationen ein Bewusstsein für die Thematik digitale Forensik nicht vorhanden. Deswegen besteht hier auch insbesondere in der Lehre und Ausbildung entsprechend Aufholbedarf, um das Wissen um die verfügbaren Mittel in die Organisationen hinein zu tragen.

Basierend auf den Ergebnissen der Studie sollten sich zukünftige Arbeiten auf dem Gebiet der digitalen Forensik in Organisationen insbesondere auf Methoden zur einfachen und möglichst kostengünstigen Implementierung von *Forensic Readiness* fokussieren. Bereits existierende Ansätze wie z.B. der Leitfaden „IT-Forensik“ [Bun11] sind teilweise sehr auf einzelne Implementierungen spezialisiert und nicht wissenschaftlich diskutiert. Gerade im Bereich der digitalen Forensik ist es jedoch nötig Methoden auch wissenschaft-

lich zu diskutieren, da zum einen die Beweiskraft durch die Verwendung anerkannter und überprüfter Best Practices gesteigert werden kann und zum Anderen auch Beschuldigte bzw. unabhängige Sachverständige die Möglichkeit haben die angewandten Verfahren transparent nachzuvollziehen.

Danksagung

Die Arbeit wurde unterstützt von "Regionale Wettbewerbsfähigkeit und Beschäftigung", Bayern, 2007-2013 (EFRE) als Teil des SECBIT Projekts (`http://www.secbit.de`).

Literatur

[ABB+12] Ross Anderson, Chris Barton, Rainer Böhme, Richard Clayton, Michael van Eeten, Michael Levi, Tyler Moore und Stefan Savage. Measuring the Cost of Cybercrime. In *11th Annual Workshop on the Economics of Information Security, WEIS 2012, Berlin, Germany, 25-26 June, 2012*, 2012.

[BSJ10] David Barske, Adrie Stander und Jason Jordaan. A Digital Forensic Readiness Framework for South African SME's. In *Information Security South Africa Conference 2010, Sandton Convention Centre, Sandton, South Africa, August 2-4, 2010. Proceedings ISSA 2010*. ISSA, Pretoria, South Africa, 2010.

[Bun11] Bundesamt für Sicherheit in der Informationstechnik. Leitfaden IT-Forensik, 2011.

[Cas11] Eoghan Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet*. Academic Press, 3. Auflage, 2011.

[CK12] Anargyros Chryssanthou und Vasilios Katos. Assessing Forensic Readiness. In *Proceedings of the Seventh International Workshop on Digital Forensics & Incident Analysis (WDFIA 2012)*, 2012.

[DF11] Andreas Dewald und Felix Freiling. *Forensische Informatik*. Books on Demand, 1. Auflage, 2011.

[FS07] Felix C. Freiling und Bastian Schwittay. A Common Process Model for Incident Response and Computer Forensics. In *IT-Incidents Management & IT-Forensics - IMF 2007, Conference Proceedings, September 11-13, 2007, Stuttgart, Germany*, Jgg. 114 of *LNI*, Seiten 19–40. GI, 2007.

[MGL11] Antonis Mouhtaropoulos, Marthie Grobler und Chang-Tsun Li. Digital Forensic Readiness: An Insight into Governmental and Academic Initiatives. In *Proceedings of the 2011 European Intelligence and Security Informatics Conference*, EISIC '11, Seiten 191–196. IEEE Computer Society, 2011.

[Nel06] Anthony Nelson. ISO 27001 as a Support to Digital Forensics. *J. Digital Forensic Practice*, 1(1):43–46, 2006.

[PIP10] George Pangalos, Christos Ilioudis und I. Pagkalos. The Importance of Corporate Forensic Readiness in the Information Security Framework. In *Proceedings of the 2010*

19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2010), WETICE '10, Seiten 12–16. IEEE Computer Society, 2010.

[PK10] Georgios Pangalos und Vasilios Katos. Information Assurance and Forensic Readiness. In *Next Generation Society. Technological and Legal Issues*, Jgg. 26 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Seiten 181–188. Springer Berlin Heidelberg, 2010.

[PMB03] John Patzakis, Scott Mann und Melisa Bleasdale. Computer Forensics in the Global Enterprise. In *1st Australian Computer, Network & Information Forensics Conference, 25 November 2003, Perth, Western Australia*. School of Computer and Information Science, Edith Cowan University, Western Australia, 2003.

[Row04] Robert Rowlingson. A Ten Step Process for Forensic Readiness. *International Journal of Digital Evidence (IJDE)*, 2(3), 2004.

[Tan01] John Tan. Forensic Readiness, 2001.

[WWW03] Jeni Wolfe-Wilson und Henry B. Wolfe. Management strategies for implementing forensic security measures. *Information Security Technical Report*, 8(2):55–64, 2003.

[YM01] Alec Yasinsac und Yanet Manzano. Policies to Enhance Computer and Network Forensics. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*. 2001.

Experiences and observations on the industrial implementation of a system to search over outsourced encrypted data.

Patrick Grofig, Martin Haerterich, Isabelle Hang, Florian Kerschbaum, Mathias Kohler, Andreas Schaad, Axel Schroepfer, Walter Tighzert

SAP AG

andreas.schaad@sap.com

Abstract: This paper presents an industrial report on the implementation of a system that supports execution of queries over encrypted data. While this idea is not new, e.g. [HILM02, AKRX04, PRZB11], the implementation in a real world large scale in-memory database is still challenging.

We will provide an overview of our architecture and detail two use cases to give the reader an insight into how we technically realized the implementation. We then provide three main contributions, reporting that:

a) We significantly improve functionality by intelligently splitting query execution, i.e. which parts of a query can be performed in the cloud and which on the client.

b) We share some initial performance measurements with the community on basis of the TPCH benchmark.

c) We present a domain-specific analysis of three data sets that shows the effects of executing queries over encrypted data and what adjustments are required with respect to the encryption of individual columns.

The three made observations on query execution, execution time measurements and domain-specific query analysis will lead us to the conclusion that although searching over outsourced encrypted data is always a trade off between functionality, performance and security, it is realistic to assume that working solutions can be provided in the not too distant future to the market.

1 Introduction

1.1 The Problem

On-demand databases outsourced in the "Cloud" are vulnerable to additional attacks compared to on-premise databases. While the cloud provider organization is usually trusted, its employees like database operators may misuse their elevated privileges to access cloud data. One recent prominent example is that where a database administrator of a Swiss Bank sold the records of clients to German and US Tax authorities. Clouds also represent a valuable target for attackers and a single vulnerability in the provider's system landscape can put all customers at risk.

1.2 The naive approach

The obvious approach could be to encrypt all data with a secure encryption algorithm such as AES and store it in the cloud. However, while secure, all data can no longer be processed in the cloud but has to be downloaded and decrypted on the client to execute any query on it. This makes any serious Database as a Service offering questionable and is the way many traditional DBMS like Sybase, Oracle, DB2 or solutions like Dropbox appear to work when they claim to encrypt data and provide cloud storage.

1.3 The Solution: Searching over Encrypted Data

However, we can provide proof that it is possible to execute arbitrary SQL queries on encrypted data without any decryption on the server-side by following three key ideas [PRZB11]:

- first, a SQL-aware encryption strategy that maps specified SQL operations to "fitting" encryption schemes (e.g. deterministic, order-preserving, homomorphic or searchable) supporting the requested functionality;
- second, an "adjustable, layered, query-based encryption" (onion encryption) that can adjust the encryption level of each item to the required level for the desired functionality;
- third, a query optimization algorithm that can improve the trade-off between security and performance as well as a set of algorithms improving columnar re-encryption required for join operations.

We developed a framework (called “SEEED” – Search over Encrypted Data) that allows to provide scalable and fine-grained encryption at a columnar-level at the same time providing the ability to directly execute queries over encrypted data. One application could be in a JDBC context, another secure JPA persistency. The framework has been prototypically implemented in the SAP HANA database management system [FMLG+12].

1.4 The Delta: True Security for Outsourced Data

Our approach provides a significant delta to the current approach of securing outsourced data. As mentioned before, just encrypting tables is not sufficient as any application would require decryption before processing. If this happens in the cloud, we are vulnerable to any attacks from the cloud provider and if the decryption and query execution happens on the on-premise client any Database as a Service offering is pointless.

More specifically, we do not only provide table encryption but encryption of individual columns paired with the ability to execute SQL directly over the encrypted columns. Our approach scales as it also allows to leave non-critical columns in cleartext without any change to a query required.

Note, that primary key material never leaves the client system. The final result set of any query processed in the cloud is sent in an encrypted fashion back to the client where it is decrypted.

The abstract use case is that a database query (eg. select, insert, update) triggered by an application is intercepted by our SEEED Database driver (eg JDBC). The SEEED driver will encrypt the query elements and send the encrypted query to the database. The encrypted query is executed over the encrypted data and the encrypted result is sent back to the application. Only the application can decrypt the result as primary encryption keys never leave the client.

The implicit trust assumption is that the client / on-premise environment is trusted, the network is untrusted and that the server / cloud environment is honest but curious. Prior to outsourcing, tables and individual columns have been encrypted following our onion approach. Data is always encrypted in a randomized fashion at the outermost encryption layer (for example, AES in CBC mode). This means that when leaving the on-premise environment, the dataset is encrypted following current security best practices.

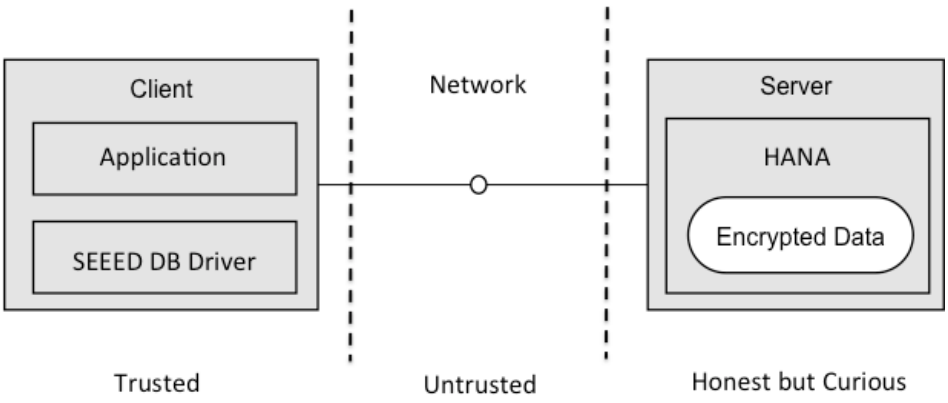


Figure 1: Simplified Data Flow Diagram

2 Architecture

2.1 Main Components

Extending the basic description provided in Section 1.4, Figure 2 now provides a more detailed description of the architectural components and their interaction introduced by SEEED. We again emphasize that the web application server hosting a business application and providing JDBC services is running in a trusted on-premise domain.

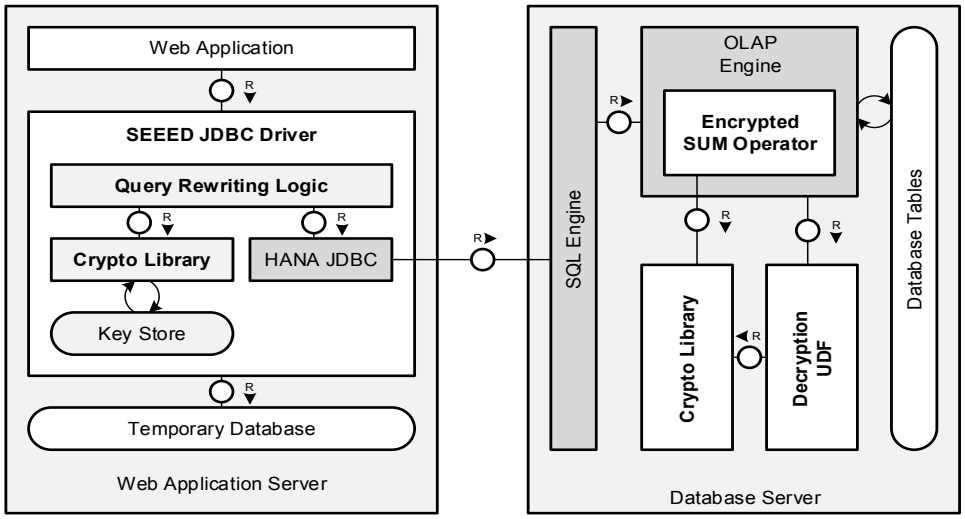


Figure 2: Architecture Overview

On the side of the Web Application Server, the SEEED JDBC Driver is the central component for connecting and querying the encrypted database. It contains a Query Rewriting Logic which transforms a plain SQL query given by the Web Application into an operator tree based on which all further processing is done. This includes encryption of single elements within the tree, selection of specific encryption schemes and their respective onions and layers, optimization of the execution sequence given the several operators in the tree, as well as the decisions which operators are executed on the server and which have to be processed on the client by means of post-processing.

Data encryption is done by a local Crypto Library which supports a range of different crypto schemes (compare section 2.3) each providing its own specific search characteristics. For key management we use a primary key provided by the Web Application based on which the secondary encryption keys for de- and encryption are stored in a local Key Store (realized by using the standard Java KeyStore implementation).

The SEEED JDBC driver uses the original HANA JDBC driver to execute the encrypted queries on HANA. For possible post processing of query results (compare section 3.1), a local Temporary Database is used.

In Figure 2, the Database Server is a SAP HANA system where a new operator for aggregations on encrypted data was implemented with the OLAP Engine. Besides many other query expressions, this extension enables a client to explicitly execute SUM operations directly on HANA. The additional crypto library on the server provides the respective cryptographic algorithms. Moreover, it also supports the Decryption UDFs which are used to update the respective encrypted data if onion layers have to be removed or re-encrypted (compare section 2.2).

2.2 Onions & Re-encryption

The two fundamental concepts of our approach are based on so called onions of encryption, i.e. cryptographic schemes that a) support specific operations over the encrypted data and b) allow transformation from one scheme to another.

One plaintext column is encrypted and stored on the database by multiple onions. Each onion consists of different layers. The center layer is always the plaintext. Each layer then encrypts the result of the previous layer with a specific encryption scheme.

For example, one possible onion may be the following: AES Randomized (AES Deterministic (Order Preserving (Cleartext Column))) as shown in Figure 3. This basically means that the plaintext of a column is first encrypted under an Order Preserving scheme, the result is then encrypted under an AES scheme in deterministic mode and that result using AES in Randomized mode.

Another onion might have a plaintext layer and only one encryption layer featuring homomorphic encryption. A third onion might have a plaintext layer and one encryption layer with searchable (SCR) encryption schema. A last fourth one might only have plaintext and an AES Randomized encryption layer (a so called “retrieval onion” as it is only used for transferring data from the server to the client).

Each layer(!) of an onion is used for a specific purpose of an SQL query. So, for simplicity, let us assume, a column c1 is currently encrypted using AES Deterministic (DET) which supports an SQL operation such as join() or equals(). If another SQL query wants to perform a range query it may be more efficient to re-encrypt c1 into an Order Preserving Scheme (OPE).

This is the core of our approach and requires careful planning about how to initially encrypt data and when to do a re-encryption at runtime.

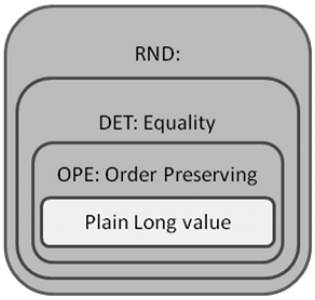


Figure 2: One possible Encryption Onion

2.3 Encryption Schemes

As part of current framework we currently use the following encryption schemes:

- Randomized: Cipher texts belonging to a sequence of plaintexts are indistinguishable from random values. Used if just the data is retrieved from the database and no operation on server side is required.

  Example: AES in CBC mode.

- Deterministic: Cipher texts belonging to same plaintexts are identical. Here security depends on the entropy of the data (for example, if there are only two distinct values, e.g. „female“ and „male“ we may face a frequency attack).

  Example: AES in ECB mode.

- Order-preserving: For ordered domains only (numerical values or lexicographically ordered texts).

  Example: Boldyreva et al. [BCLN09]

- Partially Homomorphic: Enables aggregation functions, in particular SUM()

  Example: Paillier [Pai99]

- Searchable Encryption: Used to perform exact searches without revealing anything else than the result set.

  Example: Song et al. [SWP00]

- Re-Encryption (DET-JOIN): Deterministic scheme where it is possible to change the key (re-encrypt) on without intermediary decryption. For an optimal re-encryption strategy see [KHGK+13].

  Example: Pohlig and Hellmann [PH78]

We must strongly emphasize that it is not valuable to simply „compare“ the encryption schemes in a „A is more secure than B“ fashion. As part of section 3.3 we discuss the security implications of individual schemes in the context of domain-specific data.

3 Accepting the trade-offs

It is an accepted viewpoint that searching over encrypted data is a continuous trade-off between Functionality, Security and Performance. The following sections will now present some results in the context of our implementation efforts that relate to each one of the three variables.

3.1 Functionality

Since most relational operators work without modification on encrypted data, rewriting the SQL Query becomes simple. The entire rewriting process can be performed on the abstract syntax tree. This is efficient, but also limited. Some operators, such as like() or bitwise operators, as well as some sequences of operators, such as sorting or selecting after aggregation, cannot be executed with unmodified operators. As a database developer one can now either reject these queries asking the user to rewrite or – in these rare cases – retrieve the data and execute the query locally. Of course, it is still beneficial to execute as much as possible on the server. Hence, the query needs to be split into a part that can be executed on the server and a part that must be executed at the client. Such a split can no longer be performed on the declarative SQL syntax. Instead we parse the query and construct the query in relational algebra. A tree of relational operators can then be split into a remote part on the server and a local part on the client. The leaves of the tree are database tables (scan operations). These are always executed on the server. The tree above and including the bottom most operator that can no longer be executed on the server is executing at the client. We translate each part into a SQL query that is either executed on the server on encrypted data or on the temporary database of the client on decrypted data. Intermediate tables are transferred to the client. This local/remote split of the operator tree enables executing all SQL queries, since the local database can execute any query on the decrypted, retrieved data. As such our encrypted database supports the entire SQL functionality. For a detailed description of the algorithm see [KHKH+13].

3.2 Performance

On basis of current prototypical implementation we made the following initial performance measurements (in ms). These are based on running queries for a table size of 1,000,000 rows. Both, server and client are running on HP Z820 workstations with 8 quad core CPUs and 128 GB RAM, operating SUSE Linux Enterprise Server 11 SP2.

Looking at only an exact search and only taking the server-side, i.e. the computation on the encrypted data, into consideration we can see that the impact (Factor 1.2) is marginal and the same is the case for an equi-join (Factor1.5). On the other hand, a grouping with

aggregation, i.e. sum(), operation is costly and will incur a penalty of factor 11.7. It is an open research question to identify more efficient additively homomorphic encryption schemes. Lattice-based cryptography seems to be a promising candidate.We can observe the impact of the aggregation as well in the TPCH query 5, which besides some basic selection, join and a range condition includes a sum() operation.

```
SELECT
    N_NAME, SUM(L_P_DISC_PRICE)
FROM CUSTOMER,
    ORDERS, LINEITEM, SUPPLIER, NATION, REGION
WHERE C_CUSTKEY = O_CUSTKEY AND L_ORDERKEY = O_ORDERKEY
[...]
AND O_ORDERDATE >= '1994-01-01' AND O_ORDERDATE < '1995-01-01'
GROUP BY N_NAME
ORDER BY SUM(L_P_DISC_PRICE) DESC
```

Figure 5: TPCH Query 5

Test Case		SEEED	Plain	Impact
Server-Side Only	Exact Search	2.0	1.7	1.2 x
	Equi-Join	49.7	33.3	1.5 x
	Grouping with Aggregation (Sum)	674.1	57.8	11.7 x
Incl. Client-Side	Order by Aggregate (Sum)	870.1	56.3	15.4 x
TPCH	Q4	2,402	235	10.2 x
	Q5	1,373	207	6.6 x

Figure 6: Performance measurement

3.3 Security

It makes no practical sense to reason about security just on basis of the applied encryption schemes. For example, any order-preserving scheme will be rather more subject to a statistical attack than a fully randomized scheme. What is more important is to understand the impact of queries on individual columns in terms of possible re-encryption operations as well as the domain-specific semantics of the involved data.

Example I: The TPCH Industrial Benchmark

The TPCH benchmark is a standard ERP database benchmark, with in total 61 different columns in 8 tables. Running all 22 queries of the TPCH benchmark results in the following adjustment at the individual column level: 27,90 % of all columns stay randomly encrypted and 39,30 % of all columns stay deterministic or randomly encrypted.

	TPCH	TPCC	Customer X
Total Queries	22	20	406
Total Tables	8	9	2
Total Columns	61	71	248
RND (columns / %)	17 / 27,9%	49 / 69%	157 / 63,3%
DET (columns / %)	24 / 39,3%	17 / 24%	74 / 29,8%
OPE (columns / %)	20 / 32,8%	5 / 7%	17 / 7,9%

Figure 7: Distribution of Encryption Onions

So, the interesting result is that only 32,80 % of all columns have to be encrypted with lower security encryption schemes to achieve the full functionality of the TPC-H benchmark. Some of these columns appear to be rather descriptive, i.e. P_Name or P_Brand, where it may be arguable whether even cleartext would be of any value to an attacker.

Example II: The TPCC Industrial Benchmark

The TPCC is another benchmark focused on OLTP scenarios. For the TPCC benchmark, the results are even more promising as only 7.9% of all data appear to be subject to a re-encryption to the OPE scheme and in fact 69% of all data remain in a fully deterministic encryption. Looking in more detail at the semantics of the OPE affected columns, we again find rather descriptive names such as C_First (Customer Firstname), O_ID (Order ID) or S_QUANTITY (Order Quantity).

Example III: Major International Consumer Goods Producer

The ERP database we evaluated consumes about 43 Gigabyte pure disk space. As part of our evaluation we only considered two tables with in total 248 columns. Initially, we faced 936 queries touching these tables, but could reduce these to 406 by leaving out blanks and batches resulting in the following observations: Many sum() operations included in the queries will lead to many Paillier encryptions (about 118 columns). All in all we did not encounter any complex queries, some queries requiring DET / DETJOINs but mostly “retrieval” of Randomized and Paillier Encryptions. More significantly, from 248 columns only 17 need an OPE encryption for these 406 queries. Those 17 columns again appear to be rather non-critical, eg. time of record creation, company code, business area or year.

4 Discussion

Our current research has from the beginning onwards focused on providing functionality as our customers have a business to run. We strongly believe that any solution comparable to what we propose will have to implement mechanisms to split query execution between the cloud and an on-premise environment if we want to support complex business queries. We saw that very initial performance measurements on somewhat realistic datasets appear to be acceptable in some cases, e.g. for exact search

and equi-join, while other operations such as grouping with aggregation may not be acceptable in an industrial context. However, what may be acceptable from a functional and performance perspective has to be evaluated in the context of different types of data storage scenarios, e.g. a customer requiring secure data archiving for quarterly audit purposes as opposed to a manufacturer with real-time production data analysis needs.

Regarding the data provided when analyzing three industrial data sets with respect to resulting column encryption after query execution in section 3.3, we clearly demonstrated that „security“ is domain-specific. It is up to the customer to decide and accept the risk whether an outsourced data set containing post codes may be re-encrypted using an order-preserving scheme. This of course has an impact on the administrative capabilties of our solution regarding the entire database lifecycle. When we initially encrypt data using our onion approach before outsourcing it to the cloud, we have to be able to support selective and constrained columnar encryption. Selective means that a customer may from the beginning onwards decide to a leave column unencrypted, whereas constrained implies that a column may never be re encrypted using, for example, an order-preserving scheme. This of course immediatly suggests to create domain-specific templates, e.g. for the financial or healthcare industry, which will help the customer in optimizing his configurations. At runtime, i.e. when the data has been outsourced and query execution is running in the cloud, the made configurations will impact our query process as detailed in section 3.1. For example, if a legitimate query with a range condition encounters a column that should not be re encyrpted using an order-preserving scheme, then the query plan has to rewrite the query such that the data set and subset of the query including the range condition are executed on the client.

5 Summary & Conclusion

This paper is an industrial experience report on the prototypcial implementation of a system to execute SQL over encrypted data. Our writing style reflects the “hands-on” approach but we tried to compensate for any scientific imprecision by pointing the reader to the relevant existing research body of work. We have demonstrated that the idea of adjusting the encryption levels by Popa et al. [PRZB11] works in a real-world setting and significantly improves the security compared to order-preserving only approaches [AKRX04]. Furthermore we enhanced their algorithms, such that they work for arbitrary SQL queries. While this is mostly an algorithmic effort in database design, such functionality can play a crucial role in adoption, since the application’s SQL queries do not need to be rewritten when moving to encryption. Furthermore we present the resulting performance trade-off.

The most prominent recent result, however, is that of having analysed three data sets with respect to resulting column encryptions after query execution. We clearly demonstrated that security is domain-specific. Future work will now focus on automating data set and query analysis to constrain (onion)encryption before outsourcing data to the cloud as well as defining acceptable re-encryption operations or local / remote split query strategies at runtime.

References

[AKRX04] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Order preserving encryption for numeric data. In Proceedings of the 2004 ACM International Conference on Management of Data, SIGMOD, 2004.

[BCLN09] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In Proceedings of the 28th International Conference on Advances in Cryptology, EUROCRYPT, 2009.

[FMLG+12] Franz Färber, Norman May, Wolfgang Lehner, Philipp Große, Ingo Müller, Hannes Rauhe, and Jonathan Dees, "The SAP HANA database – an architecture overview," IEEE Data Engineering Bulletin 35(1), pp. 28–33, 2012.

[HILM02] Hakan Hacigümüs, Balakrishna R. Iyer, Chen Li, and Sharad Mehrotra. Executing sql over encrypted data in the database-service-provider model. In Proceedings of the ACM International Conference on Management of Data, SIGMOD, 2002.

[KHKH+13] Florian Kerschbaum, Martin Härterich, Mathias Kohler, Isabelle Hang, Andreas Schaad, Axel Schröpfer, Walter Tighzert. An Encrypted In-Memory Column-Store: The Onion Selection Problem. In Proceedings of the 9th International Conference on Information Systems Security, ICISS, 2013.

[KHGK+13] Florian Kerschbaum, Martin Härterich, Patrick Grofig, Mathias Kohler, Andreas Schaad, Axel Schröpfer, Walter Tighzert. Optimal Re-Encryption Strategy for Joins in Encrypted Databases. In Proceedings of the 27th IFIP WG 11.3 Conference on Data and Applications Security, DBSEC, 2013.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the 18th International Conference on Advances in Cryptology, EUROCRYPT, 1999.

[PH78] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over gf(p) and its cryptographic significance. IEEE Transactions on Information Theory, 24(1):106–110, 1978.

[PRZB11] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In Proceedings of the 23rd ACM Symposium on Operating Systems Principles, SOSP, 2011.

[SWP00] Dawn X. Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In Proceedings of the 21st IEEE Symposium on Security and Privacy, S&P, 2000.

Effizienteres Bruteforcing auf einem heterogenen Cluster mit GPUs und FPGAs*

J. Fuß, B. Greslehner-Nimmervoll, W. Kastl, R. Kolmhofer

Department Sichere Informationssysteme, FH OÖ
Softwarepark 20
A-4232 Hagenberg im Mühlkreis
{juergen.fuss, robert.kolmhofer}@fh-hagenberg.at
{bernhard.greslehner-nimmervoll, wolfgang.kastl}@fh-hagenberg.at

Abstract: Heterogene Cluster mit verschiedenen Coprozessor-Typen können homogenen Lösungen überlegen sein. Diese Arbeit behandelt die Implementierung von verteilten, kryptoanalytischen Aufgaben in einer heterogenen Umgebung. Die Verwaltung der Komponenten übernimmt ein eigenständiges Cluster-Framework. Es wird verwendet, um passwortgeschützte PDF-Dokumente – verteilt auf CPUs, GPUs und FPGAs – zu brechen. Zusätzlich zeigen die Ergebnisse, dass ein Verwaltungs-System für heterogene Hardware keine beträchtlichen Leistungs-Einbußen verursachen muss.

1 Einleitung

Aufwändige kryptoanalytische Aufgaben werden üblicherweise parallelisiert gelöst – entweder in verteilten Systemen mit sehr wenig oder gar keinem Kommunikationsbedarf oder in Cluster-Knoten mit ähnlicher oder sogar gleicher Prozessorarchitektur. In den folgenden Abschnitten wird ein Ansatz vorgestellt, der verschiedene Coprozessoren verwendet, um die Vorteile eines heterogenen Systems effizient zu nutzen. Ein Vorteil eines solchen Systems besteht darin, dass unterschiedliche Aufgaben auf dem jeweils dafür optimalen System gelöst werden können. Komplexe Aufgaben können in Teilaufgaben unterteilt und verschiedenen Coprozessoren zugewiesen werden, um eine höhere Leistung zu erzielen. Dieser Ansatz kann aufgrund der Modularität zusätzlich Entwicklungszeit einsparen. In dieser Arbeit wird der Nutzen dieser Methode demonstriert, indem das Passwort eines geschützten PDF-Dokuments durch einen Brute-Force-Angriff ermittelt wird. Zuerst wird der benötigte Schlüssel auf Grafikkarten berechnet und später auf FPGAs parallel validiert.

*Diese Arbeit wurde finanziert im Sicherheitsforschungs-Förderprogramm KIRAS vom Bundesministerium für Verkehr, Innovation und Technologie. Wesentliche Teile dieser Arbeit wurden bereits in [DFG+13] veröffentlicht.

2 Themenbezogene Arbeiten

Es existieren mehrere Arbeiten, die die Benutzung von Grafikprozessoren (GPUs) und Field Programmable Gate Arrays (FPGAs) zur Beschleunigung von kryptographischen Anwendungen behandeln. In diesem Kontext von Bedeutung ist speziell [LWC+09], welches die Bruteforce-Attacke von PDF-Verschlüsselung auf GPUs beschreibt. Ergänzend existieren mehrere Ansätze [WYWS12, HMH09, WLT11], welche die Beschleunigung von MD5 auf GPUs behandeln, sowie die effiziente Implementierung von RC4 auf FPGAs [KL08]. Die Ergebnisse der genannten Arbeiten flossen in die Entwicklung des hier vorgestellten PDF-Passwort-Crackers ein. Im Projekt AXEL wird ein ähnlicher Ansatz verfolgt, um klassische High-Performance-Computing-Probleme zu lösen [TL10].

3 Entschlüsselung passwortgeschützter PDF-Dokumente

Diese Arbeit behandelt ein gängiges Verschlüsselungsverfahren des Portable-Document-Format(PDF)-Standards nach ISO 3200 [Ado08]. Es verwendet eine Kombination von MD5 und RC4 für die Verbesserung der Schlüsselsicherheit und RC4 oder AES für die eigentliche Verschlüsselung. Dies ist die Standardeinstellung für eine Vielzahl von PDF-Generatoren.

Abbildung 1 zeigt den kompletten Algorithmus in schematischer Form. Der erste Schritt besteht in der Berechnung des Encryption Keys. Dieser Schlüssel ergibt sich durch mehrfaches MD5-Hashen des Benutzerpassworts, welches mit verschiedenen Parametern aus dem PDF-Header – P-Entry, O-Entry, File-ID und ein konstanter Initialwert – kombiniert wird. Der Paramter P-Value spezifiziert, welche Zugriffrechte auf das PDF-Dokument gewährt werden. O-Entry ist ein 32-Byte-String und vom Benutzerpasswort abgeleitet. Dieser wird benutzt um die Korrektheit eines beim Öffnen des Dokuments eingebenen Passworts festzustellen. Der zeitaufwändigste Teil ist die Berechnung der 51 MD5-Runden, wovon jede den Ausgabewert der vorherigen MD5-Runde als Eingabeparameter benötigt. Der Encryption Key wird für die eigentliche Dokumentenverschlüsselung verwendet. Er ist die Basis für die Berechnung des User Keys. Der User Key wird für die Überprüfung eines eingegebenen Passworts eingesetzt. Eine Kopie dieses Schlüssels ist im PDF-Header für die Passwortvalidierung gespeichert. Wenn ein Benutzer ein Passwort zur Dokumentenentschlüsselung eingibt, wird der User Key berechnet und mit dem im Dokument gespeicherten verglichen. Sind beide identisch, wurde das richtige Passwort eingeben. Die implementierte Bruteforce-Attacke verwendet diesen Mechanismus, um das korrekte Passwort zu ermitteln. Der User Key wird berechnet, indem ein MD5-Hash der File-ID erzeugt und 20-mal mit RC4 verschlüsselt wird. Der Encryption Key dient als Eingabe. Der Hash der File-ID kann im Voraus berechnet werden, um Zeit zu sparen. In Summe benötigt eine einzelne Passwortvalidierung 51 MD5-Berechnungen und 20 RC4-Verschlüsselungen.

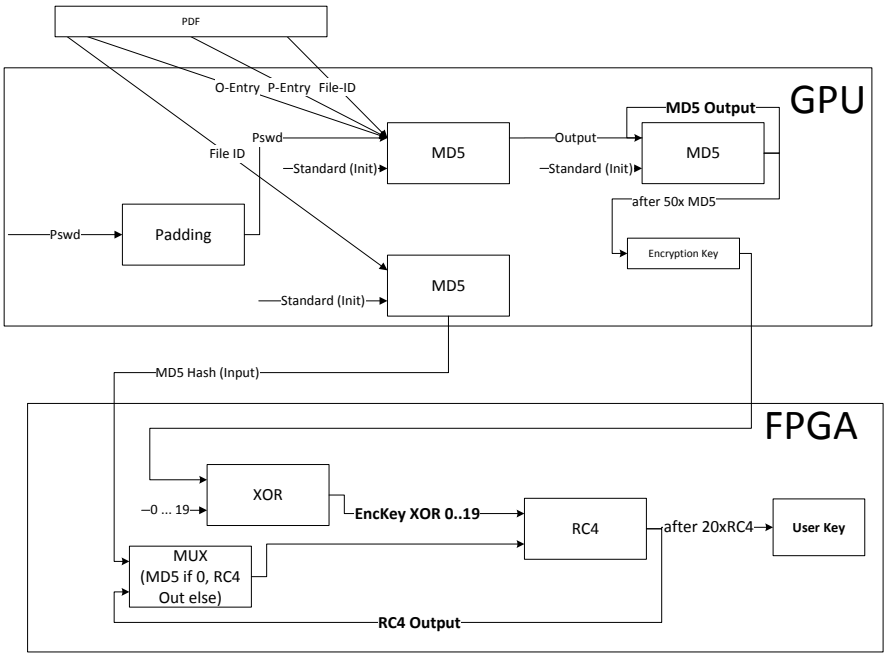


Abbildung 1: Ablauf der PDF Entschlüsselung.

4 Implementierung

Die Berechnung eines User Keys wird auf zwei verschiedenen Coprozessor-Typen ausgeführt. Die GPUs generieren die Encryption Keys mit Hilfe des MD5-Algorithmus. Die Ausgabe wird an die FPGAs weitergeleitet. Dort werden die mit RC4 verschlüsselten User Keys mit dem Schlüssel des PDF-Headers verglichen (siehe Abbildung 1). Diese Aufteilung wurde aufgrund der Eignung der jeweiligen Coprozessoren durchgeführt. Durch die wenigen Speicherzugriffe (außerhalb von Registern) ist der MD5-Algorithmus für eine Berechnung durch GPUs geeignet. Der RC4-Algorithmus hingegen ist nur mit Leistungs-Einbußen auf GPUs ausführbar. Die großen, intern im Algorithmus verwendeten 256-Byte-Arrays passen nicht in die relativ kleinen, schnellen Speichereinheiten der GPU. Daher müsste auf den größeren aber um ein Vielfaches langsameren, globalen Speicher ausgewichen werden. Dies wurde mit Messungen bestätigt: Auf einer NVIDIA-Geforce-GTX-680-Grafikkarte konnte ein Durchsatz von 17,16 Millionen Encryption Keys pro Sekunde erzielt werden. Dagegen wurden lediglich 950.000 RC4-verschlüsselte User Keys pro Sekunde erreicht.
Insgesamt erreicht die komplette GPU-Implementierung des Algorithmus einen Durchsatz von 680 kKps (Tsd. Schüssel pro Sekunde) auf einer NVIDIA-Geforce-GTX-680-Grafikkarte.

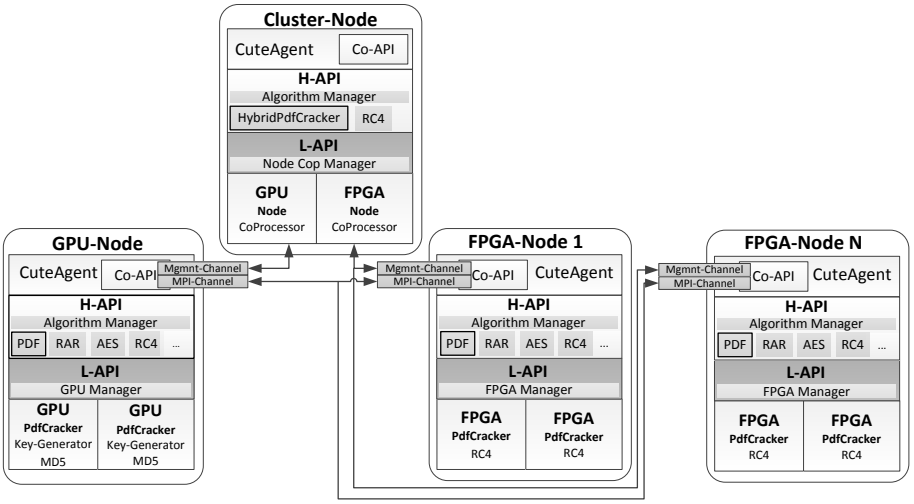


Abbildung 2: Schematischer Auftbau der Hard- und Software-Komponenten inklusive Datenfluss.

4.1 Cluster-Verwaltung

Das System verwendet ein eigenständiges Cluster-Framework um die Verteilung der Teilaufgaben und den Datenfluss durch den Cluster zu steuern (siehe Abbildung 2). Es besteht aus zwei Basiskomponenten: dem CuteMaster, welcher als Kontrollinstanz auf der Management-Node des Clusters ausgeführt wird und dem CuteAgent, welcher als Message Passing Interface (MPI)-Job auf den Compute-Nodes verteilt wird. Der CuteMaster konfiguriert das System mittels Konfigurationsdatei oder Benutzereingabe und überträgt die notwendigen Informationen durch den Management-Channel (via Transmission Control Protocol (TCP) über Ethernet) an die CuteAgents.

Sobald die Parameter – Pfad zur PDF-Datei; Passwortmaske; Anzahl der Blöcke, auf die der Passwortsuchraum aufgeteilt wird; Adressen der Clusternodes – übermittelt wurden, verbinden sich die CuteAgents mit den vorher zugewiesenen Coprozessoren, initialisieren sie und öffnen die MPI-Verbindungen für die Kommunikation über Infiniband mit den anderen Cluster-Nodes. In der ersten Phase werden Encryption Keys auf den GPU-Nodes erzeugt. Mehrere Schlüssel werden zu Schlüsselblöcken zusammengefasst und gleichmäßig auf die FPGA-Nodes verteilt. Dort wird die Berechnung mit der Generierung der User Keys fortgesetzt, welche abschließend mit dem User Key des PDF-Dokuments verglichen werden.

4.2 GPU-Teil

Eine NVIDIA GeForce GTX 680 GPU berechnet die MD5-Hashes. Diese wurde wegen ihrer Verfügbarkeit, niedrigen Hardwarekosten und fortgeschrittenen Entwicklungswerkzeuge ausgewählt. Als Programmiersprache kommt C for CUDA zum Einsatz. Der NVIDIA-Grafikkartentreiber führt in Zusammenarbeit mit der CUDA Runtime Library Speichertransfers mit einer Übertragungsgeschwindigkeit von ca. 4,6 GBps zu und von der GPU durch.

Um die Wiederverwertbarkeit des Programmcodes zu steigern, wurde die Berechnung auf den GPUs in drei Kernels aufgeteilt. Der erste Kernel generiert die Passwörter in Übereinstimmung mit einer festgelegten Passwortmaske. Der zweite übernimmt das Padding. Der dritte Kernel hasht die Passwörter samt Padding mit dem MD5-Algorithmus. So kann z. B. die Passwortgeneratorkomponente ausgetauscht werden.

4.2.1 Passwort-Generierung

Für die Passwortgenerierung wird ein einfacher Generator verwendet, welcher Passwörter auf Basis einer Passwortmaske erstellt. Diese Maske erlaubt einzelne Zeichentypen für jede Passwortposition. Z.B. ist es möglich, Passwörter zu generieren, welche mit zwei Zahlen beginnen und von vier Buchstaben und zwei Sonderzeichen gefolgt werden. Diese feine Kontrolle ist nützlich, wenn Informationen über das Passwort vorhanden sind.

Die erzeugten Passwörter werden im globalen Speicher auf der GPU so gespeichert, dass optimale Zugriffsmuster (coalesced access) angewendet werden können.

4.2.2 Padding

Die generierten Passwörter werden mit einem Padding String – dem PDF-Standard [Ado08] entsprechend – gefüllt, sodass sie exakt 32 Bytes lang sind.

4.2.3 Hashing

Aufgrund der seriellen Architektur des MD5-Algorithmus berechnet jeder GPU-Thread alleine einen kompletten MD5-Hash. Der GPU-Thread lädt das gepaddete Passwort vom globalen Speicher und hasht dieses mit dem O-Wert (Owner Key), welcher im konstanten Speicher abgelegt ist.

Der MD5-Algorithmus ist als optimiertes Macro implementiert, welches garantiert, dass oft benutzte Variablen in Registern abgelegt werden. Sogar mit dieser Maßnahme bleibt der Ressourcenverbrauch des Kernels niedrig. Wird für die neueste GPU-Generation (Compute Capability 3.0) kompiliert, so werden 30 Register und 87 Bytes *constant memory* benötigt. Der niedrige Ressourcenverbrauch ergibt ein *occupancy* von 100 %. Das bedeutet, dass der GPU-Scheduler so viele Threads wie möglich verwaltet, um Speicherzugriffsverzögerungen durch intelligentes Context-Switching zu verbergen. Der GPU-Thread holt

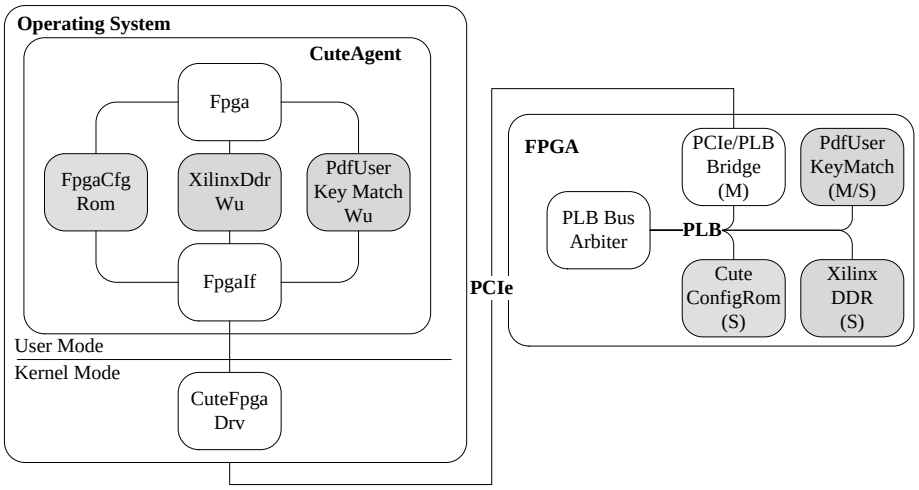


Abbildung 3: Übersicht FPGA-Node.

sich nur zu Beginn Daten aus dem langsamen, globalen Speicher. Daher kann angenommen werden, dass die GPU maximal ausgelastet ist und keine Leerlaufzeiten aufgrund von kostspieligen Speicheroperationen entstehen.

4.3 FPGA-Teil

Als Plattform wurde das Xilinx-ML605-FPGA-Entwicklungs-Board (ML605) verwendet. Die entwickelte Intellectual Property (IP) wurde im Xilinx Embedded Development Kit (EDK) als Buskomponente hinzugefügt. Aufgrund der Einfachheit und Stabilität wurde zur Kommunikation der Peripheral Local Bus (PLB) gewählt, welcher ursprünglich von IBM spezifiziert wurde. Auf den Einsatz des Advanced eXtensible Interface Bus (AXI) wurde verzichtet, da zur Zeit der Entwicklung die Unterstützung in der Xilinx Tool Chain nicht ausreichend war. Die Verbindung zwischen Host und ML605 wurde mit der Peripheral Component Interconnect Express (PCIe)-Schnittstelle realisiert.

In Abbildung 3 sind die am FPGA implementierten Teile der PDF-Entschlüsselung dargestellt. Das System besteht aus zwei Hauptblöcken, der Kontrollsoftware im CuteAgent und dem FPGA System on a Chip (SOC). Die beiden Komponenten sind via PCIe und PCIe-Treiber verbunden.

4.3.1 Infrastruktur

Die Infrastruktur wird hauptsächlich vom Softwaremodul „Fpga“ verwaltet. Dieses bewerkstelligt die Enumeration der FPGA-Boards. Die FPGA-Konfiguration wird herausge-

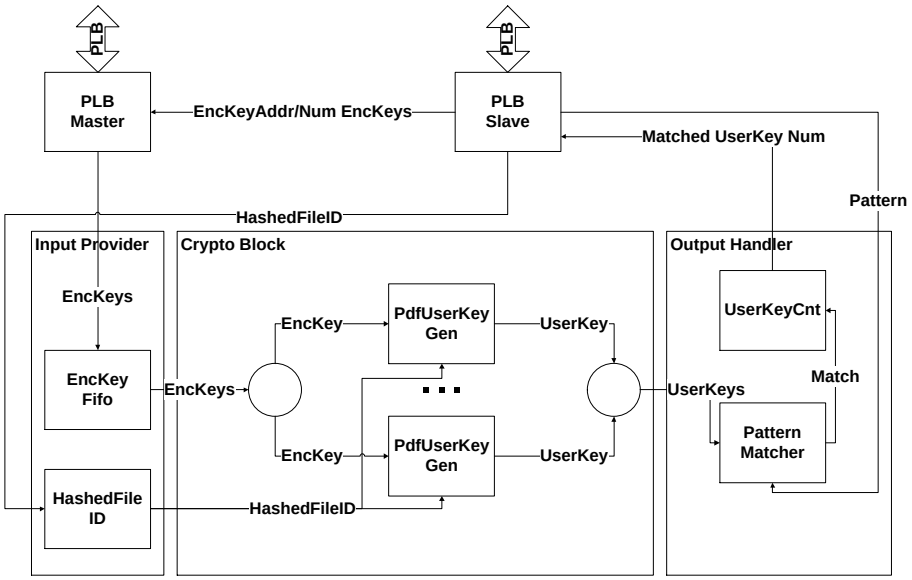


Abbildung 4: Struktur des „PdfUserKeyMatch“-IP.

funden, indem das SW-Modul „FpgaCfgRom“ die Informationen vom Hardware-IP „CuteConfigRom“ ausliest. Das Read Only Memory (ROM) enthält die Anzahl der im SOC befindlichen IPs ebenso wie deren Identifikation und SOC-Busadressen. Anschließend werden die dazugehörigen SW-Module geladen.

4.3.2 Entschlüsselungsprozess

Die FPGA-IPs für den eigentlichen Entschlüsselungsprozess sind der Xilinx Double Data Rate (DDR) Block und der „PdfUserKeyMatch“. Zuerst werden die von der GPU generierten Encryption Keys in das DDR RAM auf das ML605 transferiert. Sobald dieser Vorgang abgeschlossen ist, liest die „PdfUserKeyMatch“-IP die übertragenen Schüssel aus dem RAM und berechnet die entsprechenden User Keys. Falls der berechnete User Key mit dem in der PDF-Datei hinterlegten übereinstimmt, wird ein Flag gesetzt und der Index des gefundenen Encryption Keys gespeichert. Der Index wird benötigt, um das von der GPU verschlüsselte Originalpasswort wiederzufinden.

4.3.3 PdfUserKeyMatch-IP

Abbildung 4 zeigt die Struktur der „PdfUserKeyMatch“-IP. Es gibt drei Hauptkomponenten: Input Provider, Crypto Block und Output Handler:

Der Input Provider liest mit Hilfe des PLB Master-Blocks die Encrpytion Keys von einer gegebenen RAM-Adresse, und fügt die Schüssel ins EncKey-FIFO ein. Zusätzlich wird

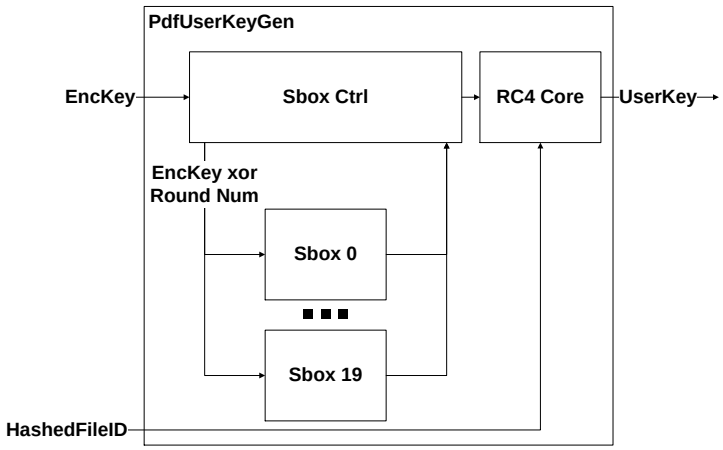


Abbildung 5: Blockdiagramm des User Key-Generators.

der MD5-Hash von der PDF File ID (HashedFileID) vor dem Entschlüsselungsprozess in einem Register gespeichert, welches später vom Crypto Block verwendet wird.

Der Crypto-Block liest die Encryption Keys und die gehashte PDF File ID vom Input Provider und verteilt die Informationen an den User-Key-Generator. Die berechneten User Keys werden zum Output Handler weitergereicht. Abbildung 5 zeigt eine Übersicht über den User-Key-Berechnungsblock. Das Ändern des Encryption Keys (Herstellen des RC4-States) benötigt den größten Berechnungsaufwand mit 1027 Zyklen. Hingegen benötigt die Ver-/Entschlüsselung für eine Runde 68 Zyklen. Deshalb sind die RC4-Zustände als getrennter Block für jede Runde implementiert. Dies erlaubt ein Pipelining des State-Initialisierungsprozesses und die Verwendung eines einzelnen RC4-Blocks für alle Runden.

Der Output Handler vergleicht die berechneten User Keys mit dem von der PDF-Datei erwarteten. Der gefundene Index „Matched UserKey Num“ kann am Schluss über die PLB-Slave-Schnittstelle gelesen werden.

4.3.4 FPGA-Durchsatz

Für die Transformation eines Encryption Keys in den entsprechenden User Key zeigt die Simulation, dass die IP 1360 Taktzyklen für die 20 RC4-Runden benötigt. Die Synthese (Xilinx EDK 14.1) ergibt für das FPGA am ML605-Board, dass 22 IPs mit einer Frequenz von 100 MHz betrieben werden können. Der theoretische Durchsatz beträgt somit 1,6 MKps (Mio. Schüssel pro Sekunde).

5 Cluster-Durchsatz

Die Leistung des gesamten Systems wurde in einem Cluster mit 16 Nodes gemessen. Die Clusterverwaltung wird auf einer separaten Node ausgeführt. Ebenso ist eine mit einer Grafikkarte ausgestattete Node (GPU-Node) im Cluster reserviert. Die restlichen 14 Nodes (FPGA-Nodes) sind jeweils mit einem FPGA-Board bestückt. Jede Node im Cluster ist mit einem 56 Gbit/s FDR-InfiniBand Host Channel Adapter (HCA) ausgestattet.

Der Durchsatz der FPGA-Implementierung wurde ermittelt, indem die Zeit zwischen dem Laden der Encryption Keys in das FPGA DDR RAM und dem Ende der Berechnung am FPGA gemessen wurde. Die beobachteten 0,6 MKps entsprechen jedoch nicht den theoretischen 1,6 MKps. Die Analyse der PLB-Transaktionen mit Xilinx ChipScope ergab, dass das Hauptproblem in der ineffizienten Implementierung der Xilinx PCIe to PLB Bridge liegt. Des Weiteren mindert auch die Verbindung vom PLB zum IP (Xilinx PLB to Intellectual Property Interface Bridge) den Durchsatz. Einen weiteren Grund für die geminderte Leistung stellt der einfache Treiber dar. Dieser bietet weder für Direct Memory Access (DMA) noch für Interrupts Unterstützung. Im Cluster arbeiten alle FPGA-Nodes mit einer Geschwindigkeit von 8,4 MKps.

Für die Durchsatzmessungen der GPU wurden die Encryption Keys berechnet, ohne sie zu den FPGAs zu senden. Die GPU generiert 17,5 MKps.

Das gesamte System berechnet 7,9 MKps, was ungefähr dem FPGA-Durchsatz entspricht. Somit kann gesagt werden, dass bei dieser Anwendung durch das Cluster-Framework keine nennenswerten Einbußen bei der Gesamtperformance des Systems verursacht werden.

Als Vergleich dient eine Implementierungsvariante, die auschließlich mit GPU-Nodes realisiert wurde (siehe Abschnitt 4). Jede GPU-Node ist mit einer NVIDIA GeForce GTX 680 GPU ausgestattet. Da jede GPU den kompletten Algorithmus (inkl. RC4) ausführt, ohne Zwischenergebnisse an andere Coprozessoren weiterzureichen, wird keine Kommunikation benötigt. Der Schlüsselraum muss lediglich auf alle Nodes aufgeteilt werden. Eine einzelne GPU-Node erreicht einen Durchsatz von 680 kKps. 15 GPU-Nodes erzielen somit 10,2 MKps.

6 Ausblick und Verbesserungen

Der Durchsatz der heterogenen Implementierung liegt rund 20 % unter der der homogenen GPU-Implementierung. Der Flaschenhals ist die PCIe-Anbindung des FPGAs. Seit der Veröffentlichung von [DFG+13] wurden die Host-FPGA-Schnittstelle und das FPGA-Design überarbeitet. Für die schnelle Übertragung der Encryption Keys vom Host zum FPGA wurde eine Direct-Memory-Access-Unterstützung (DMA) hinzugefügt. Damit stellt das Laden der Encryption Keys zum FPGA keinen Engpass mehr dar. Im neuen Design steuert der DMA-Controller am FPGA den Datenfluss. Damit können die Encryption Keys ohne den Umweg über das DDR RAM vom PCIe-Block zum „PdfUserMatch“-IP übertragen werden.

Des Weiteren wurde das System auf Interrupts umgestellt, damit die Bandbreite der PCIe-Schnittstelle nicht durch Polling (Prüfen des Done-Bits) belastet wird.

Die Taktfrequenz der Kryptoblöcke kann mit dem Synthese-Tool (Xilinx ISE 14.4) nicht erhöht werden, da der kritsche Pfad im unüberarbeiteten RC4-Block liegt. Aufgrund der Ressourcen-schonenden FPGA-Schnittstelle lassen sich anstatt der 22 RC4-Cores zusätzlich zwei Crypto-Cores instanzieren. Damit ergibt sich ein theoretischer Schüsseldurchsatz von 1,76 MKps. Die Messung auf einer Node ergeben einen Durchsatz von 1,73 MKps. Das überarbeitete Design erreicht somit annähernd eine dreifache Steigerung gegenüber der alten Implementierung.

Ausführliche Tests und Leistungsmessungen im Custer sind noch ausständig. In Summe darf mit diesen Maßnahmen eine signifikate Durchsatzsteigerung für das Gesamtsystem erwartet werden, welche die Leistung der GPU-Variante deutlich übersteigt.

Literatur

[Ado08] Adobe Systems Incorporated. Document management — Portable document format — Part 1: PDF 1.7, 2008.

[DFG+13] B. Danczul, J. Fuss, S. Gradinger, B. Greslehner-Nimmervoll, W. Kastl und F. Wex. Cuteforce Analyzer: A Distributed Bruteforce Attack on PDF Encryption with GPUs and FPGAs. In *Proceedings of the Eighth International Conference on Availability, Reliability and Security (ARES), 2013*, Seiten 720–725. IEEE Computer Society Conference Publishing Services, 2013.

[HMH09] Guang Hu, Jianhua Ma und Benxiong Huang. High Throughput Implementation of MD5 Algorithm on GPU. In *Proceedings of the 4th International Conference on Ubiquitous Information Technologies Applications, 2009. ICUT '09*, Seiten 1–5, 2009.

[KL08] S.H.M. Kwok und E.Y. Lam. Effective Uses of FPGAs for Brute-Force Attack on RC4 Ciphers. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 16(8):1096–1100, 2008.

[LWC+09] Changxin Li, Hongwei Wu, Shifeng Chen, Xiaochao Li und Donghui Guo. Efficient implementation for MD5-RC4 encryption using GPU with CUDA. In *3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication, 2009. ASID 2009*, Seiten 167–170, 2009.

[TL10] Kuen Hung Tsoi und Wayne Luk. Axel: a heterogeneous cluster with FPGAs and GPUs. In *Proceedings of the 18th annual ACM/SIGDA international symposium on Field Programmable Gate arrays*, FPGA '10, Seiten 115–124, New York, NY, USA, 2010. ACM.

[WLT11] Hongwei Wu, Xiangnan Liu und Weibin Tang. A fast GPU-based implementation for MD5 hash reverse. In *Anti-Counterfeiting, Security and Identification (ASID), 2011 IEEE International Conference on*, Seiten 13–16, 2011.

[WYWS12] Feng Wang, Canqun Yang, Qiang Wu und Zhicai Shi. Constant memory optimizations in MD5 Crypt cracking algorithm on GPU-accelerated supercomputer using CUDA. In *2012 7th International Conference on Computer Science Education (ICCSE)*, Seiten 638–642, 2012.

Sicherheitskonzept zum Schutz der Gateway-Integrität in Smart Grids

Carl-Heinz Genzel[1], Richard Sethmann[2], Olav Hoffmann[3], Kai-Oliver Detken[4]

[1,2 und 3] Hochschule Bremen, Flughafenallee 10, 28199 Bremen
[4] DECOIT GmbH, Fahrenheitstraße 9, 28359 Bremen

carl-heinz.genzel@hs-bremen.de, sethmann@hs-bremen.de,
olavhoffmann@googlemail.com, detken@decoit.de

Abstract: Um den Herausforderungen zukünftiger Energienetze begegnen zu können, ist eine sichere Datenübertragung zwischen den Steuerkomponenten notwendig. Das Bundesamt für Sicherheit in der Informationstechnik (BSI) hat hierfür Sicherheitsvorgaben in Bezug auf eine zentrale Kommunikationseinheit, das sog. Smart Meter Gateway, (SMGW) entwickelt. Das Sicherheitskonzept berücksichtigt diese Vorgaben und erhöht zusätzlich die Informationssicherheit des SMGW indem es Elemente des Trusted Computings (TC)-Ansatzes integriert. Dazu wird ein Tamper-Resistant-Grid über ausgewählte Hardwareelemente gelegt, das Secure Boot-Verfahren angewendet und eine laufende Integritätsmessung des SMGWs über den Trusted Network Connect (TNC) - Ansatz realisiert.

1 Einleitung

1.1 Zukünftige Energienetze

Zukünftige Energienetze stehen vor den Herausforderungen schwankende und dezentrale Energieerzeugung zu ermöglichen und gleichzeitig die Netzstabilität zu wahren. Weiterhin gilt es, verschiedenste Externe Marktteilnehmern (EMT) mit ihren Interessen zu berücksichtigen [Bu13a, S. 14]: Den Messstellenbetreiber (MSB), der verantwortlich für die Messsysteme ist, den Messdienstleister (MDL), der das Ab- und Auslesen von Verbrauchszähleinrichtungen übernimmt, den Verteilnetzbetreiber (VNB), der das örtliche Stromnetz unterhält und wartet, den Lieferanten, der als Handelswarenvertreter auftritt und für die Nutzung des Netzes Gebühren an den VNB zahlt sowie den SMGW-Administrator (GWA), der in viele wesentliche Prozesse des SMGW-Lebenszyklus eingebunden ist (Datenübertragung, Administration und Eichung im laufenden Betrieb) [Bu13a, S. 13], [Bu13f S. 9, 138]. Das hier vorgestellte Sicherheitskonzept wurde im Rahmen des Forschungsprojektes „**S**ichere **P**owerl**i**ne-**D**atenkommunikation im intelligenten **E**ne**r**gienetz“ (SPIDER) erarbeitet und hat sowohl eine hohe praxistaugliche als auch gesellschaftliche Relevanz. Diese Sichtweise wird von dem Energieversorger Vattenfall Europe Distribution Berlin GmbH geteilt, sie schreiben:

„…Wir sehen in den Forschungs- und Entwicklungszielen von SPIDER und der Zusammenarbeit mit dem Konsortium die Möglichkeit die eigenen Produkte und Dienstleistungen an die steigenden Sicherheitsanforderungen im Bereich der sicheren Datenübertragung in Energienetzen anzupassen und daraus wichtige Impulse für aktuelle und zukünftige Geschäftsfelder zu gewinnen…. Wir bescheinigen den Projektzielen eine hohe gesellschaftliche Relevanz und sehen in dem geplanten Vorhaben einen hohen Innovationsgrad…“

1.2 Definition und Beschreibung des Szenarios

Die neuen Anforderungen an Energienetze können nur durch die Koordination der Energieerzeugung und des Energieverbrauchs, in Verbindung mit einer sicheren Datenübertragung zwischen den Beteiligten, erreicht werden. In diesem Zusammenhang werden zwei neue Komponenten in intelligenten Energienetzen benötigt, das Smart Meter (SM) als „Intelligenter Zähler“ und das SMGW als zentrale Kommunikations-einheit. Sie bilden zusammen die Basis des Smart Metering Systems.

Die dargestellten Komponenten und Bereiche in Abb. 1 sowie die entsprechenden Sicherheitsanforderungen in einem Smart Metering System werden durch Vorgaben des BSIs im Detail beschrieben (vgl. [Bu13a], [Bu13b], [Bu13d], [Bu13e]).

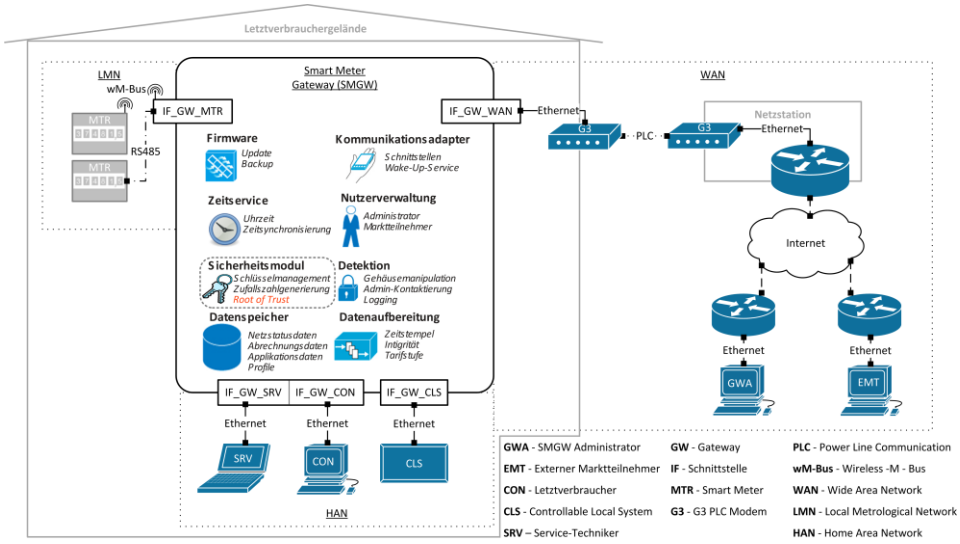


Abbildung 1: Szenario: SMGW-Kommunikation über PLC mit WAN

Das SMGW ist die zentrale Instanz in einem Smart Metering System, es besitzt die Logik zur verlässlichen Verarbeitung und sicheren Speicherung von Messdaten angeschlossener Messsysteme und soll die sichere Datenübertragung zwischen den einzelnen Teilnehmern in den angeschlossenen Netzen ermöglichen. Bei den Netzen handelt es sich gem. den Vorgaben des BSI (vgl. [Bu13a, S. 13-15]) um:

- Das Local Metrological Network (LMN), ein Netz zur lokalen Anbindung von Messgeräten (Strom-, Gas- oder Wasserzähler) der Endnutzer (Letztverbraucher (CON, LV)).

- Das Home Area Network (HAN), ein Netz zur lokalen Anbindung und Steuerung von Energieerzeugern und Energieverbrauchern (Controllable Local Systems (CLS)) der Letztverbraucher sowie zur Informationsbereitstellung für Letztverbraucher und technisches Betreiberpersonal (Service-Techniker (SRV)).

- Das Wide Area Network (WAN), zur Anbindung des GWA für die SMGW-Verwaltung und Dritter (EMT) zur Datenvermittlung.

Das SMGW erfüllt außerdem die Funktion einer Firewall zur Separierung dieser Netze und deren Teilnehmer. Neben dieser logischen Trennung sind alle Netze zusätzlich physikalisch voneinander getrennt [Bu13a, S. 13-15].

Ein Security Modul innerhalb des SMGWs stellt kryptographische Operationen für die sichere Speicherung und Übertragung von Daten zur Verfügung. Zu den Funktionen zählen unter anderem:

- Sichere Speicherung von Zertifikats- und Schlüsselmaterial

- Schlüsselgenerierung und Schlüsselaushandlung auf Basis von elliptischen Kurven

- Erzeugung und Verifikation digitaler Signaturen

- Zuverlässige Erzeugung von Zufallszahlen

[Bu13b, S. 10]

Das SMGW empfängt über das angeschlossene LMN Messwerte von Smart Metern. Smart Meter unterscheiden sich von regulären Messsystemen insbesondere dadurch, dass sie eine kryptographisch gesicherte Kommunikation zum SMGW verwenden und die Übermittlung von Messwerten durch das SMGW steuerbar ist [Bu13a, S. 15-16].

Um eine möglichst einfache Integration des Smart Meter Systems zu ermöglichen, werden zusätzliche Komponenten zur Anbindung des SMGWs an das Weitverkehrsnetz verwendet. Jedes SMGW kommuniziert mit Hilfe der G3 Power Line Communication (PLC) Technologie über die „Last Mile“ des lokalen Stromnetzes mit der nächsten Netzstation. Erst in der Netzstation wird die Kommunikation in ein vorhandenes Weitverkehrsnetz eingeleitet [SHB13 S. 333].

Welche Daten in das Weitverkehrsnetz kommuniziert werden dürfen, ist ebenfalls durch das BSI in den entsprechenden Richtlinien geregelt. In diesem Zusammenhang definiert das BSI Eigentumsverhältnisse in Bezug auf die einzelnen Rollen. Der Letztverbraucher, als natürliche oder juristische Person, ist Eigentümer der Messwerte und davon

abgeleiteter Daten seiner Messsysteme. Ein EMT ist Interessent und Nutzer dieser Daten, sie ermöglichen ihm die Durchführung der Bilanzierung, Tarifierung und Netzzustandserfassung. Der GWA hat im Allgemeinen keinen Zugriff auf diese Form der Daten. Er hat im Gegenzug Zugriff auf systemrelevante Daten wie Konfigurationsdaten, System- und Eichtechnische-Logs. Der Service-Techniker hat eine Diagnosefunktion und darf daher systemrelevante Daten auslesen, sie aber im Gegensatz zum GWA nicht speichern. Der Zugriff auf das SMGW ist jedem Teilnehmer nur über das ihm zugeordnete Netz aus Abb. 1 gestattet [Bu13a, S. 118-119].

1.3 Bedrohungsanalyse

Die Bedrohungen in dem betrachteten Szenario können laut BSI (vgl. [Bu13d, S. 33]) in folgende Kategorien aufsteigend nach Tiefe der Bedrohung eingeteilt werden:

1. Aufdecken von Daten, die sich fest auf dem SMGW oder in der Verarbeitung durch das SMGW befinden (Verbrauchsdaten, Zählerstände, Profile etc.). Ziel: Informationsbeschaffung aus der Smart Meter Infrastruktur.

2. Manipulation von Daten, die sich fest auf dem SMGW oder in der Verarbeitung durch das SMGW befinden (Tarifdaten, Tarifprofile, Zählerstände etc.). Ziel: Änderungen zum eigenen Vorteil oder zur Störung des Betriebs.

3. Veränderung und Kontrolle beteiligter Systeme (CLS, SMGW etc.). Ziel: Beeinträchtigung der Infrastruktur.

Jede Bedrohung kann zusätzlich nach Angriffsort unterschieden werden. Dabei sind Angreifer aus dem WAN als höher motiviert zu betrachten, während Angreifer aus dem HAN geringer motiviert sind [Bu13d, S. 33].

Die Bedrohungen wurden durch den STRIDE-Ansatz [Mi13] analysiert. STRIDE steht für **S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of service und **E**levation of privilege. Die zugehörigen Sicherheitseigenschaften sowie die entsprechenden Gegenmaßnahmen in Verbindung mit Trusted Computing und den BSI-Vorgaben sind in [SHB13] und [Be13] beschrieben.

2 Stand der Technik

2.1 Vertrauenswürdiges Bootverfahren

Es ist mit vergleichsweise hohem Aufwand verbunden, Hardwarebausteine einer Plattform auszutauschen oder zu manipulieren, da sie oft mechanisch gesichert sind. Die Manipulation von Software ist dagegen verhältnismäßig einfach. Aus diesem Grund ist der Schutz der Softwareintegrität, insbesondere bei der Bereitstellung von sicherheitsrelevanten Funktionen, durch Nachweisbarkeit notwendig. Aus der

Anforderung ergibt sich jedoch eine zyklische Abhängigkeit, da für den Nachweis meist wieder Software benötigt wird [LSW10, S. 569, 570].

Zur Unterbrechung dieser Abhängigkeit wird unter anderem beim Trusted Computing das Konzept „Root of Trust“ verwendet. In der Literatur wird der „Root of Trust“ als eine unbestreitbare Charakteristik oder Eigenschaft einer einzelnen Person oder Sache, die ihre Vertrauenswürdigkeit rechtfertigt, charakterisiert (vgl. [Ki06 S. 31]). Dadurch, dass der „Root of Trust“ nicht oder nur mit sehr hohem Aufwand verändert werden kann, bildet er die Basis für die Evaluierung einer Plattform bzw. eines Systems. Die Trusted Computing Group (TCG) beschreibt hierzu die „Chain of Trust“. Sie definiert die Integrität einer Plattform als eine Vertrauenskette. Diese Kette wird ausgehend von einem zentralen Punkt, dem „Root of Trust“, über verschiedene hierarchisch angeordnete Komponenten der Plattform während des Systemstarts gebildet. Eine Komponente (N) innheralb dieser Kette kennt und prüft hierfür den unversehrten Zustand der jeweiligen Folgekomponente (N+1) anhand festgelegter Attribute und protokolliert das Ergebnis. Geht man also davon aus, dass der „Root of Trust“ nicht einfach verändert werden kann, können in Abhängigkeit dazu auch alle Folgekomponenten nicht verändert worden sein, wenn alle Prüfungen erfolgreich waren. Während des Systemstarts können hierdurch Manipulationen (z.B. durch Fremdeinwirkung) an Hard- und Software ermittelt werden [Is09, S. 4-7], [Tr07, S. 7 - 8].

Dieser Vorgang wird auch als vertrauenswürdiger Bootprozess bezeichnet. Er wird in der Literatur in drei Kategorien unterteilt (vgl. [Sm05, S. 50]), die jedoch zum Teil synonym verwendet werden:

- Trusted Boot: Prüfung der Komponenten durch Analyse und Messung.
- Secure Boot: Prüfung der Komponenten durch Analyse und Messung inklusive festgelegter Aktionen bei negativem Prüfungsergebnis.
- Authenticated Boot: Prüfung der Komponenten durch Analyse und Messung abhängig von verschiedenen Szenarien. Festgelegte Aktionen bei negativem Prüfergebnis sind möglich. Die Szenarien beschreiben verschiedene valide Systemzustände.

2.2 Trusted Computing und die Anwendung des TNC-Schichtenmodells

Die TCG ist eine Standardisierungsorganisation aus der Industrie, die Spezifikationen für Trusted Computing entwickelt. Ihr Ziel ist es, einen offenen und hersteller-unabhängigen Standard für Trusted-Computing-Bausteine und Softwareschnittstellen zu spezifizieren. Sie sollen Veränderungen an IT-Plattformen erkennen und sowohl externe Softwareangriffe, als auch Veränderungen der Konfiguration, Sicherheitslücken oder schadhafte Anwendungsprogramme ausmachen [Tr12]. In der Informationssicherheit ist es häufig schwierig einem System (bspw. einem SMWG) zu vertrauen. Es ist oft nicht möglich zu erkennen, ob das Gerät (bezogen auf die Hard- und Software) manipuliert wurde. Bei der Lösung dieses Problems ist ein reiner Softwareansatz nicht zuverlässig genug, da Software leicht manipuliert werden kann.

Die TCG hat daher das Trusted Platform Modul (TPM) als ein zusätzliches Hardwaremodul spezifiziert. Als Basis des Vertrauens dient ein festes Schlüsselpaar innerhalb des Moduls. Der private Schlüssel verlässt das Modul nie und stellt seine Identität dar. Aufgrund der Annahme, dass der private Schlüssel nur dem TPM bekannt ist, kann durch den Einsatz von Signaturen der Ursprung des Inhalts zum System des TPMs zugeordnet werden. Eine nähere Beschreibung ist in [SHB13] zu finden.

TNC stellt Methoden zur Feststellung der Integrität von Endpunkten bereit, die als Basis für eine vertrauenswürdige Kommunikation dienen und kann dafür Funktionen des TPMs nutzen. Die aktuelle TNC-Architektur ist von der TCG in der Spezifikation 1.5 (Revision 3) vom Mai 2012 veröffentlicht worden. Die im Rahmen des Konzepts verwendeten Komponenten der TNC-Architektur werden in [SHB13] erläutert.

2.3 Vergleich der Trusted Computing-Technologie mit BSI-Mindestanforderungen

Das TPM ist ein zentraler Bestandteil im Trusted Computing und stellt die Identität eines Systems dar. In einem SMGW ist dagegen das Security Module ein zentraler Bestandteil der Identität. Beide Module besitzen dazu nicht auslesbare private Schlüssel (vgl. [Tr12, S. 1], [Bu13b, S. 56]) und sind fest in ihr umgebendes System integriert. Zudem müssen sie physikalischer Manipulation in Grenzen widerstehen können [Tr07, S. 47], [Ba06, S. 12, 30].

Neben einer festen Identität wird im Trusted Computing ein Integritätsnachweis in Form einer Integritätsmessung und einer davon abhängigen Attestierung im Rahmen von TNC verwendet. Dazu wird der Systemzustand anhand ausgewählter Systemattribute, meist durch ein TPM, gemessen und manipulationssicher gespeichert. Die Integritätsmessung wird beim Start des Systems oder auch zu speziellen Ereignissen während des Betriebs ausgeführt. Durch Remote Attestation kann das System aus der Ferne mit Hilfe der Messwerte auf ungewollte Veränderungen geprüft werden [Tr07, S. 8-10]. Das BSI schreibt in diesem Zusammenhang Selbsttests zur Verifizierung der Sicherheitsfunktionen und Daten vor [Bu13d, S. 38, 79]. Die Ergebnisse der Selbsttests sind ohne eine vertrauenswürdige Basis jedoch nicht vertrauenswürdig. Im Trusted Computing wird mit dem vertrauenswürdigen Bootverfahren aus Abschnitt 2.1 eine solche Basis geschaffen. Das BSI schreibt keine solche Vertrauensbeziehung vor.

Durch die Integritätsmessung und Attestierung können Hard- und Softwaremanipulationen erkannt werden. Das reduziert die Möglichkeiten des Angreifers, ein SMGW dauerhaft zu übernehmen. Dieser Aspekt wird in den aktuellen Spezifikationen des BSI nur pauschalisiert betrachtet. Durch die Integritätskontrolle wird zusätzlich die Authentizität der übertragenen Daten gestärkt, da unabhängig von einer PKI-basierten Authentifizierung der Zustand des SMGWs überwacht wird. Das TNC-Konzept der TCG in Verbindung mit einem vertrauenswürdigen Bootverfahren stellt daher den stärksten Sicherheitsgewinn durch Trusted Computing im Vergleich zu den BSI-Vorgaben dar. Ein aktueller TPM-Chip nach Spezifikation 1.2 kann in diesem Zusammenhang jedoch nicht verwendet werden, da die Anforderungen an die kryptographischen Algorithmen des BSIs nicht erfüllt werden. Die aktuell entstehende TPM-Spezifikation 2.0 muss diesbezüglich nach der Fertigstellung noch geprüft werden.

3 SMGW-Integrität im Smart Grid

Ausgehend von den Erkenntnissen aus Abschnitt 2 ist das folgende Konzept zur Sicherung der SMGW-Integrität im Forschungsprojekt SPIDER entwickelt worden.

3.1 Sicherstellung der Hardware

Das SMGW wird von außen wie bei aktuellen Messsystemen üblich mit einem Siegel oder einer sogenannten Plombe geschützt. Diese Komponenten dürfen während des Betriebs nicht beschädigt werden. Das SMGW besitzt zusätzlich Mechanismen, die das Öffnen des Gehäuses elektronisch erkennen. Zudem ist auf dem Hostcontroller des SMGWs ein sogenanntes „Tamper-Resistant-Grid“ aufgebracht, das eine elektronische Erkennung von Manipulationsversuchen an der Hardware ermöglicht. Alle Komponenten der Plattform sind fest fixiert und können nicht ohne weiteres entfernt werden.

3.2 Sicherstellung der Basisintegrität über Secure Boot

Für die Sicherstellung der Basisintegrität eines SMGWs soll ein Secure Boot-Verfahren verwendet werden. Ein Secure Boot-Verfahren kann, im Gegensatz zum Trusted Boot-Verfahren, ohne TPM durch vorhandene Technologien wie einen Co-Prozessor oder der aktuell in ARM-CPUs vorhandenen Trustzone [Ar13] leichter umgesetzt werden [LSW10, S. 572]. Bei der Umsetzung wird das in [LSW10, S. 570] definierte Secure Boot-Muster aus Abb. 2 angewendet.

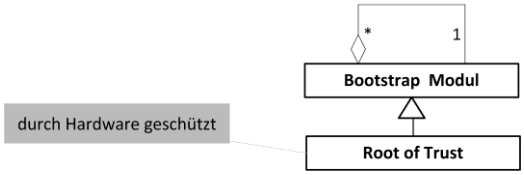


Abbildung 2: Secure Boot-Pattern gem. [LSW10, S. 570]

Der Bootprozess ist als eine Abfolge von Bootstrap Modulen gestaltet, die miteinander assoziiert sind. Das Bootstrap Modul „Root of Trust“ bildet den Ausgangspunkt des Bootprozesses und ist als eigenständiges Hardwaremodul besonders geschützt. Daraus resultiert die in Abb. 3 dargestellte Bootsequenz.

Nach dem Einschalten eines SMGWs wird zuerst das System des „Root of Trust“ aus dem Hardware-ROM geladen. Das System hat eine Referenz zum eigentlichen Bootloader (Bootstrap Modul N) und besitzt zudem eine Signatur, die den SOLL-Zustand des Bootloaders beschreibt sowie den für die Verifizierung der Signatur benötigten öffentlichen Schlüssel. Bevor das System den Bootloader lädt, wird der IST-Zustand des Bootloaders mit der SOLL-Signatur verglichen. Nur wenn die Prüfung positiv ist, wird der Bootloader geladen und die Kontrolle im Bootprozess an ihn übergeben. Der Bootloader prüft daraufhin die Hardwareintegrität (z.B. Zustand des

„Tamper-Resistant-Grid“) und das Betriebssystem (Bootstrap Modul N+1) auf dieselbe Weise mit Hilfe von Signaturen der SOLL-Zustände. Das Betriebssystem kann wiederum einzelne Applikationen (Bootstrap Modul N+M) prüfen.

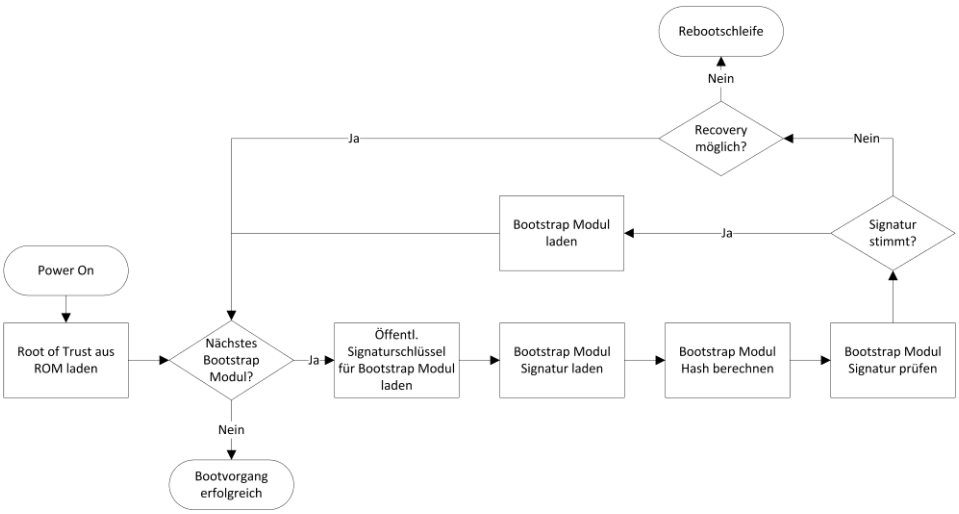


Abbildung 3: Secure Boot-Verfahren

Schlägt eine Prüfung fehl, wird der Bootvorgang unterbrochen und das System geht in einen Fehlerzustand, gekennzeichnet durch einen dauerhaften Reboot, sofern es nicht auf eine vertrauenswürdige Betriebsstufe (Recovery Möglichkeit) zurückfallen kann. Hierzu dient eine Backuppartition mit einem Duplikat der SMGW-Firmware. Sollte eine Prüfung erst oberhalb des Bootloaders fehlschlagen, ist es möglich, mit Hilfe des Bootloaders die Backuppartition für die weitere Bootsequenz zu verwenden. Erst wenn auch die Bootstrap Module auf dieser Partition nicht ihren jeweiligen SOLL-Zuständen entsprechen, bleibt das System in dem beschriebenen Fehlerzustand. Somit kann sichergestellt werden, dass es nur dann zum Betrieb eines SMGWs kommt, wenn der Initialzustand vertrauenswürdig ist.

3.3 Prüfung des SMGWs durch laufende Integritätsmessung

Es wurde festgestellt, dass der Einsatz von TNC einen signifikanten Sicherheitsgewinn darstellt. Davon ausgehend, dass die TNC-Architektur als erweiterbare Architektur beschrieben ist, kann TNC im Allgemeinen am SMGW eingesetzt werden. Der Fokus bei der Umsetzung von TNC liegt in der Ergänzung der BSI-Vorgaben durch die Integritätssicherung, während die Authentifizierung nach bestehenden BSI-Vorgaben realisiert wird.

Abb. 4 zeigt das SMGW als Network Access Requestor (NAR) und den GWA als Network Access Authority (NAA). Im Rahmen der Realisierung wird ein Integrity Measurement Collector (IMC) entwickelt. Ein TPM existiert aus den in Abschnitt 2.3 beschriebenen Gründen nicht.

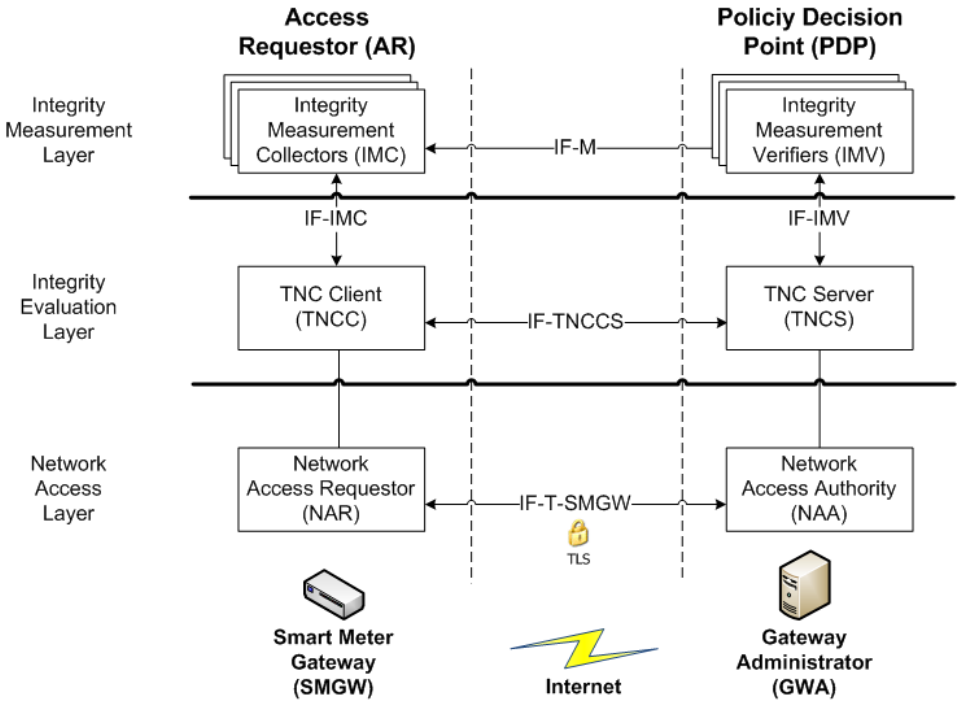


Abbildung 4: TNC-Schichtenmodell mit relevanten Komponenten des Systemkonzepts (in Anlehnung an [Tr12, S. 13] und [SHB13] Abb. 2)

Der IMC wertet Sicherheitsaspekte aus, die die Integrität des SMGWs messbar machen. Hierfür sind Hash-Summen vorgesehen, die periodisch über ausgesuchte Komponenten (z.B. eingesetzte Firmware-Komponenten, Konfigurationsdateien, Hardware-komponenten etc.) gebildet werden. Die Messwerte werden auf Dateiebene gespeichert und mit Hilfe der Mehrbenutzerfähigkeit und der granularen Dateisystemberechtigungen von Linux vor Veränderungen geschützt. Da die Dateisystemrechte auf Kernelebene geprüft werden, sind die Zugangsrechte nur schwer auszuhebeln. Im Sinne von TNC übermittelt der IMC die Messwerte zur Attestierung an den Integrity Measurement Verifier (IMV), der sich auf der Seite des GWA befindet. Dementsprechend, wird auf der Seite des GWAs ein IMV umgesetzt, der die Werte des IMCs interpretieren kann. TNC-Client (TNCC) und TNC-Server (TNCS) sind für die Kommunikation und die Reaktion auf die Ergebnisse der Attestierung zuständig. Sie liegen als standardisierte Komponenten bereits in entsprechenden Bibliotheken vor. Bei negativen Ergebnissen muss zusätzlich der GWA eingreifen. Durch die softwarebasierte Umsetzung kommt es in besonderem Maße darauf an ein System zu nutzen, dass die Integrität der Software bereits beim Systemstart verifizieren kann, um das Vertrauen in die Messwerte zu sichern.

In Abb. 4 wird der Vermittlungskanal zwischen NAA und NAR als IF-T-SMGW dargestellt, da an diesem Punkt keine bestehenden Spezifikationen der TCG für IF-T

verwendet werden kann. Die Bezeichnung IF-T-SMGW soll deutlich machen, dass die Möglichkeit besteht, an dieser Stelle eine neue Spezifikation zu erwirken. Für die Übertragung der Integritätsmesswerte vom SMGW zum GWA zur Verifizierung der Integrität, wird ein Webservice verwendet, der im Rahmen der BSI-Vorgaben für die Alarmierung und Ereignisvermittlung in Verbindung mit dem Systemzustand eines SMGWs verwendet werden soll. Alle weiteren Vorgaben zur Kommunikation über die WAN-Schnittstelle (vgl. [Bu13a, S. 22]) haben weiterhin Bestand. Um reguläre Ereignisse und Alarmierungen von TNC-Nachrichten zu unterscheiden, werden letztere speziell gekennzeichnet. Dieses Vorgehen, ermöglicht die Interoperabilität zu nicht TNC-fähigen Endpunkten. TNC-Nachrichten sind für diese Endpunkte normale Ereignisse, während TNC-fähige Geräte die Nachrichten gesondert interpretieren können. Die darüber liegenden Ebenen sind vollständig in Software umgesetzt und von den BSI-Vorgaben kaum beeinflusst, daher können auch die vorhandenen Spezifikationen verwendet werden.

4 Fazit und Ausblick

Die relevanten Aspekte zur Verbesserung der Sicherheit durch Trusted Computing sind die Integritätsmessung am SMGW und die damit verbundene Attestierung der Messwerte beim GWA in Verbindung mit TNC, da solche Überlegungen bei den aktuellen BSI-Spezifikationen bisher keine Rolle spielen. Es ist hierbei besonders wichtig die Integritätsmessung sicher durchzuführen, da sonst kein Vertrauen in die Messwerte möglich ist. Das beschriebene Sicherheitskonzept erfüllt diese Anforderung, indem eine Vertrauenskette erzeugt wird. Hierzu werden Integritätsmessungen während des Bootvorgangs (Secure Boot) und zur Laufzeit eingesetzt und die Messwerte zu den Hard- und Softwarekomponenten manipulationssicher gespeichert. Die Einbettung von Trusted Computing stellt daher einen wirklichen Mehrwert dar, um Smart Grid-Infrastrukturen wirkungsvoll absichern zu können.

Zukünftig kann der Einsatz eines Monitoring-Systems die Informationssicherheit im Smart Grid weiter erhöhen. Der Metadata Access Point (MAP) aus den TNC-Spezifikationen definiert hierfür bereits Schnittstellen, die zur zentralen Informationssammlung eingesetzt werden können. Ein Ansatz für weitere Forschung zu dem aktuellen Thema der ganzheitlichen IT-Sicherheit im Smart Grid-Umfeld.

Danksagung

Die Autoren danken dem BMWi-ZIM [Bu13g] für die Förderung und allen SPIDER-Projektpartnern [Sp13] für die gute Zusammenarbeit.

Literaturverzeichnis

[Ar13] ARM Ltd: TrustZone. http://www.arm.com/products/processors/technologies/trustzone/index.php, Nov. 2013; zuletzt aufgerufen am 22.11.13.

[Ba06] Bare, J. C.: Attestation and Trusted Computing. University of Washington, Washington, 2006.

[Be13] Becker, C: Bedrohungsanalyse für Smart Grids und Anpassung des Sicherheitskonzeptes. Hochschule Bremen, Bremen, 2013.

[Bu13a] Bundesamt für Sicherheit in der Informationstechnik: Technische Richtlinie BSI TR-03109-1 Anforderungen an die Interoperabilität der Kommunikationseinheit eines intelligenten Messsystems. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[Bu13b] Bundesamt für Sicherheit in der Informationstechnik: Technische Richtlinie BSI TR-03109-2 Smart Meter Gateway Anforderungen an die Funktionalität und Interoperabilität des Sicherheitsmoduls. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[Bu13c] Bundesamt für Sicherheit in der Informationstechnik: Technische Richtlinie BSI TR-03109-4 Smart Metering PKI - Public Key Infrastruktur für Smart Meter Gateways. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[Bu13d] Bundesamt für Sicherheit in der Informationstechnik: Protection Profile for the Gateway of a Smart Metering System (Smart Meter Gateway PP). Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013

[Bu13e] Bundesamt für Sicherheit in der Informationstechnik: Protection Profile for the Security Module of a Smart Meter Gateway (Security Module PP). Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[Bu13f] Bundesamt für Sicherheit in der Informationstechnik: Technische Richtlinie BSI TR-03109-1 Anlage VI : Betriebsprozesse. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2013.

[Bu13g] Bundesamt für Wirtschaft und Technologie: Zentrales Innovationsprogramm Mittelstand (ZIM). http://www.zim-bmwi.de/, Nov. 2013; zuletzt aufgerufen am 22.11.13.

[Is09] ISO/IEC: ISO/IEC 11889-1 Information technology — Trusted Platform Module — Part 1: Overview. ISO copyright office, Genf, 2009.

[Ki06] Kinney, S.: Trusted platform module basics : using TPM in embedded systems. Elsevier, Amsterdam [u.a], 2006.

[LSW10] Löhr, H.; Sadeghi, A.-R.; Winandy, M: Patterns for Secure Boot and Secure Storage in Computer Systems. Availability, In IEEE: ARES '10 International Conference on Reliability, and Security, Krakow, 2010; S.569-573.

[Mi13] Microsoft: Threat Modeling Uncover Security Design Flaws Using The STRIDE Approach. http://msdn.microsoft.com/en-us/magazine/cc163519.aspx, Jan. 2013; zuletzt aufgerufen am 6.11.13.

[SHB13] Sethmann, R.; Hoffmann, O.; Busch, S.: Sichere Datenübertragung in Smart Grids mit Trusted Computing. DACH-Security, Nürnberg, 2013; S.332-343.

[Sm05] Smith, S. W: Trusted Computing Platforms : Design and Applications. Springer, New York, 2005.

[Sp13] SPIDER: Partner. http://www.spider-smartmetergateway.de/cms/front_content.php?idcat=3&lang=1, Nov. 2013; zuletzt aufgerufen am 22.11.13.

[Tr07] Trusted Computing Group: TCG Specification Architecture Overview. TCG PUBLISHED, Beaverton, 2007.

[Tr11] Trusted Computing Group: TPM Main - Part1 Design Principles. TCG PUBLISHED, Beaverton, 2011.

[Tr12] Trusted Computing Group: TCG Trusted Network Connect TNC Architecture for Interoperability. TCG PUBLISHED, Beaverton, 2012.

IT-Sicherheitsaspekte industrieller Steuerungssysteme

Christian Freckmann[(1)], Ulrich Greveler[(2)]

[(1)]TÜV-IT GmbH
Bereich Informationssicherheit
Langemarckstr. 20
45141 Essen

[(2)]Hochschule Rhein-Waal
Labor für IT-Sicherheit
Friedrich-Heinrich-Allee 25
47475 Kamp-Lintfort

C.Freckmann@tuvit.de
mail@ulrich-greveler.de

Abstract: Zum automatisierten Messen, Steuern und Regeln und zur Produktionsüberwachung kommen in vielen industriellen Bereichen sogenannte Industrial Control Systems (ICS) zum Einsatz. Die Verbreitung standardisierten IT-Komponenten und die zunehmenden Vernetzungen der Systeme setzen die Systeme heute ähnlichen Gefährdungen aus wie in der klassischen Informationstechnik. Dabei haben ICS abweichende Anforderungen an die Schutzziele Verfügbarkeit, Integrität und Vertraulichkeit. Dies äußert sich beispielsweise in längeren Laufzeiten und Wartungsfenstern. Die von den Autoren gewonnenen Erfahrungen aus realen ICS-Anwendungsfällen und -Sicherheitsaudits sind in die Erstellung eines ICS-Security-Kompendiums eingeflossen, das vom Bundesamt für Sicherheit der Informationstechnik herausgegeben wird.

1 Einführung

Zum automatisierten Messen, Steuern und Regeln von Abläufen, beispielsweise zur Automation von Prozessen und zur Überwachung von großen industriellen Produktionssystemen, kommen in vielen Bereichen sogenannte Industrial Control Systems (ICS; deutsch: industrielle Steuerungssysteme, Automatisierungssysteme) zum Einsatz. Diese finden häufig Verwendung in der produzierenden Industrie und in Bereichen, die zu den kritischen Infrastrukturen gezählt werden, z. B. Energie, Chemie, Versorger, Infrastruktur-Einrichtungen.

ICS waren in der Vergangenheit physikalisch voneinander und im Hinblick auf die sie umgebende Umwelt isoliert und damit insbesondere vor äußeren Einflüssen geschützt. Aus diesem Grunde blieben IT-Sicherheitsbetrachtungen bei der Auswahl und Entwicklung der zumeist proprietären Software und Protokolle unberücksichtigt.

Die Verbreitung standardisierten IT-Komponenten und die zunehmenden Vernetzungen der Systeme setzen die Systeme heute ähnlichen Gefährdungen aus wie in der klassischen Informationstechnik., so dass die gesamte Sicherheitskonzeption von

Systemen zur Prozesssteuerung zu überdenken und der aktuellen Bedrohungslage anzupassen ist.

Abweichend zu IT-Infrastrukturen, wie wir sie aus Rechenzentren und dem Büroumfeld kennen, haben ICS spezifische Anforderungen an die Schutzziele Verfügbarkeit, Integrität und Vertraulichkeit. Augenfällig sind hier beispielsweise erheblich längere Gesamtlaufzeiten, die Nichtberücksichtigung automatisierter Systemupdates und eine sehr geringe Zahl von Wartungsfenstern. Aus diesen Gründen und aufgrund von Echtzeitanforderungen und Gewährleistungsansprüchen sind etablierte Schutzmaßnahmen aus dem Büroumfeld, wie der IT-Grundschutz für den normalen Schutzbedarf, nur bedingt übertragbar.

2 Vorfälle aus jüngerer Zeit, Angreiferstrukturen

Wir führen einige in der jüngeren Vergangenheit bekannt gewordenen Vorfälle auf, um die Bedrohungslage von ICS zu illustrieren.

Aurora-Projekt Idaho (2007): Im Rahmen des von der Idaho National Laboratories durchgeführten Projekts wurden ICS-Sicherheitseigenschaften mit kontrollierten Experimenten untersucht: Die Ventilsysteme einer Anlage bestehend aus Dieselgeneratoren eines Energieversorgers konnten von den (hier im staatlichen Auftrag handelnden) Angreifern so weit desynchronisiert werden, dass eine Verpuffung einen der Generatoren zerstörte. [CNN 2011]

Sobig-Amtrak (2009): Der Sobig-Virus befiel Systeme mit Microsoft-Betriebssystem, die im CSX-Hauptquartier in Jacksonville (Florida) für die Steuerung der Signalisierung von Amtrak-Eisenbahnzügen eingesetzt wurden. Für Teile des Netzes war damit die Signalisierungsfunktion außer Betrieb, so dass einige Züge für mehrere Stunden gestoppt werden mussten. [CBS 2009]

Stuxnet (2010) ist eine Malware, die im Juni 2010 öffentlich bekannt wurde und speziell Systeme befällt, die zur Überwachung von Siemens-Simatic-S7-Steuerungen eingesetzt werden. Stuxnet griff dabei in die Steuerung von Frequenzumrichtern der Hersteller Vacon und Fararo Paya ein, wodurch insbesondere die Leittechnik der Urananreicherungsanlage in Natanz, Iran, beeinflusst wurde. [ISIS 2010]

Hackereinbruch bei Telvent (2012): Der Hersteller von ICS für den Einsatz in Smart Grids gab im September 2012 bekannt, dass es Hackern gelungen ist, in das Firmennetz einzubrechen und Zugriff auf ICS-Projektdateien zu gelangen. Unter anderem war davon das OASyS-DNA-System betroffen, das Telemetriedatenaustausch und Kontrollfunktionen für Energieversorgungsnetze bereitstellt aber auch beim Betrieb von Öl- und Gaspipelines zum Einsatz kommt. [WIRED 2012]

Systematische behördliche und wissenschaftliche Unterscheidungen (vgl. [GAO 2004], [GAO 2005], [EPSR 2011] und [BSI Expo 2012]), die auf beobachteten Straftaten

basierten und Motivationen der Täter kategorisierten, zeigen grobe Täterschichtungen auf, die in Tabelle 1 zusammengefasst sind.

Tätergruppe	***Vorgehensweise / Motivation***
Cyber-Kleinkriminelle	Monetäres Interesse; keine Fokussierung auf besondere Industriebereiche
Ausländische Dienste	Nutzung von aufwändigen Werkzeugen; Informationsbeschaffung allgemein; Industriespionage
(kriminelle) Hacker, „Scriptkiddies“	Interesse unterschiedlich: Demonstration von besonderen Fähigkeiten; Prahlerei innerhalb einer Peer Group; Ausprobieren von Angriffswerkzeugen bei weniger versierten Tätern ohne konkretes Ziel; gezielte Imageschädigung von Großunternehmen
(politische) Hacker, „Hacktivisten“	politisches Interesse; Bekanntmachung einer Botschaft; öffentlichkeitswirksame Angriffe bzw. Einbrüche werden bevorzugt; oft stehen eher Webserver als produktive Systeme der Industrieunternehmen im Vordergrund, da diese Angriffe stärker wahrgenommen werden
Informationskrieg Infowar	Gezielter Angriff auf kritische Infrastruktur seitens eines Staates bzw. einer regierungsnahen Organisation; Unterbrechung der Versorgung mit Elektrizität, Wasser, Gas oder Verkehrs- und Kommunikationsdiensten (Telefonie, Internet, Funknetze); Sabotage militärischer Anlagen oder Kommunikationssysteme
Insider	Aktive oder ehemalige Mitarbeiter oder Lieferanten, die sich aggressiv gegenüber dem Unternehmen verhalten und unter Nutzung von Interna Sabotage initiieren oder Informationen stehlen; oft Einzeltäter mit beträchtlichem Schadenspotential
Autoren von Malware	Überschneidet sich teilweise mit kriminellen Gruppen (z. B. Verschlüsselungstrojaner); betrifft aber auch unspezifisch erstellte Malware bzw. Computer-Viren, die ICS-Systeme befallen
Terroristen	Zerstörung bzw. Unbrauchbarmachung kritischer Infrastrukturen; Gefährdung nationaler Sicherheit; Gefährdung von Menschenleben bis hin zu Massenmorden; Schädigung von Umwelt und Wirtschaft; Verbreitung von Angst und Schrecken
Industriespione	Stehlen von Entwurfsunterlagen, Know-how und Angebotskalkulationen im Auftrag von Diensten oder Konkurrenten oder mit dem Ziel, die erbeuteten Informationen an Letztgenannte zu verkaufen

Tabelle 1: Angreiferstrukturen

3. Beobachtungen bei ICS-Sicherheits-Audits

In der Tabelle 2 sind beispielhaft typische, in jüngerer Vergangenheit bei ICS-Sicherheits-Audits unter Mitwirkung der Autoren festgestellte Beobachtungen aufgeführt, welche Rückschlüsse auf die aktuelle Gefährdungslage in deutschen Industriebetrieben zulassen.

Komponente	***Sicherheitsrelevante Beobachtungen***	***Branche***
Netz	• Anbindung unbekannter Systeme zur Datensicherung	Energieversorger
Firewall/Router	• Regeln nicht restriktiv • undokumentierte Regeleinträge • nicht mehr benötigte Datenflüsse • Bypass im Routing • IP-Forwarding auf Servern	Chemie Automobilzulieferer Maschinenbau
Modems	• ungeschützter Zugang • Anschluss nicht dokumentiert • ständige Verbindung	Energieerzeuger
Fernwartung	• Anschluss direkt in Feldebene	Wasserversorgung Chemie
Betriebssysteme/ Härtung	• Betriebssystemkomponenten nicht gehärtet • ungenutzte Dienste angeboten • Nicht-unterstütze Version und fehlende Patches	Netzbetrieb (Energie) Maschinenbau Windpark
Funkverbin-dungen	• fehlende Verschlüsselung • veraltete Netzelemente	Öl/Gas-Industrie
Industrie-Switche	• fehlende Robustheit gegen unerwartete Kommunikation	Öl/Gas-Industrie Energieversorger
veraltete Netzelemente	• Administrativer, webbasierter Zugang ohne Absicherung • Fehlende Protokollunterstützung (z. B. nur 'telnet'-Zugang)	Automobilzulieferer Pharma

Tabelle 2: Ausgewählte Audit-Beobachtungen

Eine Auswertung dieser Audit-Ergebnisse legt den Schluss nahe, dass die festgestellten Schwachstellen unabhängig von Industriezweigen verbreitet sind und eher aufgrund einer allgemeinen Verbreitung von Standard-IT-Komponenten innerhalb von Produktionslandschaften Einzug in die Steuer- und Kontrollsysteme gehalten haben.

4. Besondere Angriffsvektoren, die im ICS-Umfeld verwendet werden

Die im Folgenden benannten Angriffsvektoren und Szenarien stellen typische Kombination von Angriffstechniken und Schwachstellen dar, die im ICS-Umfeld eine Rolle spielen. Hier werden für den Angreifer Erfolg versprechende und damit praktisch relevante Kombinationen von Schwachstellen und Techniken zur Ausnutzung von Schwachstellen in ICS-Umgebungen benannt.

Malware auf Steuerungs-PC mit Durchgriff auf SPS

Zur Programmierung von SPS (speicherprogrammierbare Steuerungen) in ICS-Installationen wird gewöhnlich ein eigener Rechner mit Steuerungssoftware verwendet. Es existiert auf ICS zugeschnittene Schadsoftware, welche von einem solchen infizierten Steuerungs-PC aus unbemerkt eigene Steuerungsbefehle zur Programmierung der SPS sendet. Hierbei weicht die visualisierte Darstellung auf dem Steuerungs-PC von der tatsächlichen Programmierung der SPS ab, da die veränderten Steuerungsbefehle nicht von dem vorgesehenen Programm übermittelt, sondern vor dem Absenden durch die Schadsoftware modifiziert werden. Somit hat der Prozessingenieur keine Möglichkeit diese Abweichung zu bemerken.

Für Angreifer ist dieser Angriffsvektor besonders „wertvoll“, da hierdurch nicht nur die SPS kompromittiert und die Produktion auf eine gewünschte Weise gestört wird. Es wird gleichzeitig die Visualisierung des Steuerungszustands im Sinne des Angreifers beeinflusst. In der Folge bemerkt das Bedienpersonal die Auswirkung des Angriffs nicht, schöpft keinen Verdacht und setzt die Produktion unvermindert fort. Beeinträchtigte Systeme können dann über einen langen Zeitraum sabotiert werden, ohne dass dies bemerkt wird.

Soziale Netze und Spear-Phishing in Bezug auf das ICS-Bedienpersonal

Nutzt das ICS-Bedienpersonal öffentliche Kommunikationsplattformen im Internet wie soziale Netze (z. B. Facebook, Xing, Linkedin), so kann ein Angreifer aus scheinbar belanglosen Informationen (z. B. Facebook-Nachricht über den Schichtbeginn oder über den Leerlauf des Personals aufgrund von Produktionsstopps oder einer Störung) ggf. Rückschlüsse auf die Produktion ziehen, die für gezielte Angriffe genutzt werden können.

So lassen sich beispielsweise mithilfe von sozialen Netzen Personen identifizieren, die sich mit dem Betrieb von ICS befassen. Daraufhin kann ein Angreifer nun gezielt Informationen über dieses ICS-Personal abrufen und dafür nutzen, um zugeschnittene Phishing-Angriffe (siehe Kapitel 3.3.2.6) auf einen kleinen Kreis oder nur eine Person

durchzuführen. Diese Phishing-Angriffe mittels gezielter Informationsbeschaffung werden als Spear-Phishing bezeichnet.

Während ursprüngliche Phishing-Attacken z. B. über massenversendete Emails aufgrund eines allgemeinen Sicherheitsbewusstseins an Bedeutung verlieren ([Vi2011], [Sh2011]), können diese gezielten, mit Hintergrundinformationen angereicherten Spear-Phishing-Angriffe eine vergleichsweise hohe Erfolgsquote verzeichnen.

Beispielsweise wird der Phishing-Versuch mittels eines vermeintlichen Hinweises eines Lieferanten zur Absendung eines vorgegebenen Kommandos im Fall einer Störung nicht mehr ignoriert werden, wenn tatsächlich kurz zuvor an diesen Lieferanten eine Störungsmeldung herausgegangen ist.

Das soziale Netz kann auch von Angreifern genutzt werden, um z. B. eine Freundschaftsanfrage eines vertrauenswürdigen Kollegen vorzutäuschen und anschließend diese Vertrauensstellung auszunutzen. So kann im Namen des Kollegen eine scheinbar plausible Handlung erbittet werden (z. B. kurzzeitiges Ändern eines Passworts auf eine vorgegebene Zeichenkette für einen angeblichen Test), um den Angriff auf das System durchzuführen.

Verfälschung von Sensordaten zu Sabotagezwecken

Historisch bedingt sind die im ICS-Umfeld verwendeten Netztechnologien und Protokolle nicht unter dem Gesichtspunkt der IT-Sicherheit entwickelt worden und weisen somit häufig keine Sicherheitsmechanismen auf. Für die Übermittlung von Steuerungsdaten werden häufig entsprechend ungesicherte Netztechnologien wie Profibus und DNP eingesetzt. Daher ist der Zugang zum Bus für einen Angreifer oftmals ausreichend, um die übertragenen Daten im Klartext lesen und frei verändern zu können. Die Kommunikation wird dabei nicht zwangsläufig unterbrochen oder gestört.

Kennzeichnend ist bei diesem Angriffsvektor, dass nur eine am Bus operierende Komponente (z. B. eine SPS) kompromittiert werden muss, um die Operation aller anderen Komponenten zu stören.

So können beispielsweise Sensordaten (z. B. Füllstand, Temperatur, Druck) verfälscht werden, um Abschaltungen oder Regelungen zu verhindern und damit den Produktionsprozess zu beeinflussen. Denkbar ist auch das Verfälschen von Produktionsparametern (z. B. Frequenzen, Umdrehungen, Dauer eines Schweißvorgangs), um gezielt Fehlproduktionen zu verursachen. Unter Umständen werden die falschen Produktionsparameter erst bei der Qualitätskontrolle bemerkt, da in der Visualisierung die dargestellten Parameter nicht mit den tatsächlich eingestellten Parametern durch den Angreifer übereinstimmen.

Darüber hinaus lassen sich ggf. Safety-Mechanismen zur wirkungsvollen Verhinderung einer Sabotage (z. B. Selbstabschaltung bei Überschreitung eines Drucks oder Unterschreitung eines Füllstandes) umgehen.

Physischer Angriff zur Provokation administrativer Eingriffe

Je nach Einsatzfeld der ICS-Installation kann ein Angreifer eine der Komponenten (z. B. externer Sensor oder Aktor) physisch manipulieren, um eine Reaktion der Bedienmannschaft zu provozieren. Auf diese Weise kann ein Angreifer gewisse Aktionen wie die Durchführung von administrativen Tätigkeiten beeinflussen und dann beispielsweise für weiterführende Angriffe nutzen.

So kann z. B. ein Temperatursensor erhitzt werden, um einen Alarm auszulösen und in der Folge eine gewisse Reaktion des Bedienpersonals hervorzurufen. Dies ist beispielsweise dann möglich, wenn der Angreifer annehmen kann, dass ein Wartungszugriff erfolgt, bei dem ein Passwort unsicher übertragen wird (Mitschneiden des Passwortes an einer Netzkomponente), ein Steuerbefehl (z. B. Neustart oder Schnellabschaltung) abgesetzt wird, den er für einen späteren Replay-Angriff benötigt oder ein ungesicherter Fernwartungszugang aktiviert wird, weil die provozierte Störung das Eingreifen eines Lieferanten-Mitarbeiters erfordert und er diesen Zugang dann für sich selbst nutzen kann.

Der Angriffsvektor kombiniert daher das Wissen über das produktive System selbst mit vorhandenen Schwachstellen von ICS-Komponenten.

Eine ähnliche Vorgehensweise stellt die ständige Alarmierung dar. Der Angreifer löst wiederholt eine Alarmierung aus (z. B. Unterbrechung einer Lichtschranke). Kommt es mehrfach zu einem administrativen Eingriff ohne eine Ursache identifizieren zu können, so wird ggf. vom Bedienpersonal von einer Fehlalarmierung ausgegangen und der Alarm bis auf weiteres deaktiviert. Auf diese Weise kann der eigentliche Angriff vorbereitet werden, der denselben Alarm auslösen würde.

5. Das ICS-Security-Kompendium

Die von den Autoren gewonnenen Erfahrungen aus realen ICS-Anwendungsfällen und ICS-Sicherheitsaudits sind in die Erstellung eines Kompendiums eingeflossen, das als Handreichung für mit der Sicherheit von ICS befassten Personen mit unterschiedlichen fachlichen Hintergründen dienen soll und auf dessen Grundlage Verbände und Organisationen spezifische Sicherheitsanforderungen erarbeiten können..

Das vom Bundesamt für Sicherheit in der Informationstechnik herausgegebene Kompendium [BSI13] richtet sich primär an Unternehmen und Personen, die ICS einsetzen. Es werden Grundlagen über die IT-Sicherheit von Automatisierungssystemen vermittelt und Vorgehen beschrieben, das IT-Sicherheitsniveau dieser ICS zu optimieren. Eine weitere Zielgruppe sind Unternehmen und Personen, welche die IT-Sicherheit von Automatisierungssystemen prüfen und bewerten. Zudem soll sie als Anregung für alle Personen dienen, die sich in irgendeiner Weise mit der IT-Sicherheit von Automatisierungssystemen beschäftigen.

Im Kompendium wird eine fundierte Einführung in die Grundlagen von ICS gegeben und die sicherheitsspezifischen Grundlagen von ICS werden erläutert. Neben der allgemeinen Einführung in Schwachstellen und Angriffsvektoren erfolgt eine Erläuterung der Besonderheiten von ICS, die an ICS-Anwender und IT-Sicherheitsexperten gerichtet ist.

Ein weiteres Kapitel gibt einen Überblick über nationale und internationale Organisationen und deren Standards und Quasi-Standards im Bereich der ICS-Sicherheit. Zudem werden architekturelle, technische und organisatorische Maßnahmen zum Schutz von ICS definiert und es erfolgt eine Gegenüberstellung der Best Practices zu etablierten Standards wie IEC 62443 und VDI/VDE 2182. Die Best Practices adressieren in erster Linie Betreiber von ICS.

Schließlich wird im Kompendium eine Methodik für die Durchführung von Audits in ICS beschrieben und aktuelle Trend aus dem ICS-Umfeld betrachtet.

Literaturverzeichnis

[BSI Expo 2012] Bundesamt für Sicherheit in der Informationstechnik: Cyber-Sicherheits-Exposition, Version 1.00, 15.10.2012 (S. 2)

[BSI13] Freckmann, Greveler et al.: ICS-Security-Kompendium. Bundesamt für Sicherheit in der Informationstechnik, 2013.

[CNN 2011] Ahlers, Mike M.: Inside a government computer attack exercise, http://edition.cnn.com/2011/10/17/tech/innovation/cyberattack-exercise-idaho/index.html, abgerufen am 10.05.2013

[CBS 2009] Niland, Marty: Virus Disrupts Train Signals, http://www.cbsnews.com/2100-205_162-569418.html, abgerufen am 10.05.2013

[EPSR 2011] Fovinoa, Igor Nai et al.: Cyber security assessment of a power plant, Electric Power Systems Research 81, 2011 (S. 518-526)

[GAO 2004] Government Accountability Office: Critical Infrastructure Protection: Challenges and Efforts to Secure Control Systems (GAO-04-628T), http://www.gao.gov/assets/120/110816.pdf, abgerufen am 13.05.2013

[GAO 2005] Government Accountability Office: Critical Infrastructure Protection: Department of Homeland Security Faces Challenges in Fulfilling Cybersecurity Responsibilities, http://www.gao.gov/assets/250/246516.pdf, abgerufen am 13.05.2013

[ISIS 2010] Albright, David; Brannan, Paul; Walrond, Christina: Did Stuxnet Take Out 1,000 Centrifuges at the Natanz Enrichment Plant?, http://isis-online.org/uploads/isis-reports/documents/stuxnet_FEP_22Dec2010.pdf, abgerufen am 10.05.2013

[WIRED 2012] Zetter, Kim: Maker of Smart-Grid Control Software Hacked, http://www.wired.com/threatlevel/2012/09/scada-vendor-telvent-hacked/, abgerufen am 10.05.2013

[Vi2011] Vishwanatha et al.: Why do people get phished? Testing individual differences in phishing vulnerability within an integrated, information processing model. Decision Support Systems, Volume 51, Issue 3, June 2011, p. 576–586

[Sh2011] Shashidhar, Chen: A phishing model and its applications to evaluating phishing attacks. Proceedings of the 2nd International Cyber Resilience Conference 2011. p. 63-69

Security-Awareness-Programm unter Berücksichtigung viraler Marketingmethoden

Kirsten Brox, Anastasia Meletiadou
Unternehmenssicherheit und Compliance
buw Holding GmbH
Rheiner Landstraße 195
49078 Osnabrück
kirsten.brox@buw.de
anastasia.meletiadou@buw.de

Abstract: In diesem Beitrag wird ein Programm zur effektiven Sicherheitssensibilisierung im Unternehmen entwickelt. Schwerpunkte des Programms liegen auf Bedrohungsszenarien, der Form eines ‚Alternate Reality Game' zur Wissensvermittlung und messbaren Erfolgsgrößen. Mithilfe des Programms soll eine dauerhafte Verhaltensänderung erreicht werden.

1 Zielsetzung

Unwissenheit und mangelndes Verantwortungsbewusstsein in Bezug auf Informationssicherheit führen zu erheblichen Problemen für Unternehmen und Behörden. Die sicherste Informationstechnik kann ihren Schutz nicht entfalten, wenn Mitarbeiter mit den überlassenen Informationen fahrlässig agieren. In den vergangenen Jahren haben Studien wiederholt gezeigt, dass eine der größten Schwachstellen in der Informationssicherheit der Mensch ist: Mitarbeiter die Kennungen weitergeben, sich nicht gut über Informationssicherheit informiert fühlen oder den Wert ihres Unternehmenskennworts unterschätzen [Ros12] [Fon12] [DGHA12].

Zudem gibt es Bereiche, in denen Informationssicherheit nicht ausreichend oder nicht wirtschaftlich durch technische Lösungen erreicht werden kann. Beispiele sind etwa der Schutz vor Phishing-Angriffen oder einer gezielten Ansprache von Mitarbeitern, um sie zur Preisgabe vertraulicher Informationen zu bewegen. Es ist anzunehmen, dass mit erhöhtem Aufkommen von Angriffen auch die Zahl der erfolgreichen Versuche ansteigt. Deshalb ist es nicht verwunderlich, dass beispielsweise die Zahl der versuchten Phishing-Angriffe von 2011 bis 2013 um 87% zugenommen hat [ZAO13].

Die beiden zentralen Elemente von Awareness-Programmen sind sowohl den Mitarbeitern die erforderlichen Wissensinhalte zu vermitteln, als auch den inneren bzw. äußeren Anreiz zu schaffen, dieses Wissen anzuwenden. Im Rahmen dieses Beitrags wird Security-Awareness als ein Gesamtprogramm von Sensibilisierungselementen zur nachhaltigen Verhaltensänderung definiert. Ziele sind:

1. Mitarbeiter sollen notwendige Sicherheitsmaßnahmen und Regeln als sinnvoll erkennen und positiv besetzen. Angestrebt wird eine Einstellungsänderung vom ‚Regeln einhalten müssen' hin zum ‚Vorbeugen können'.
2. Es ist darauf zu achten, dass Maßnahmen unternehmenskulturell passend implementiert werden und von den Rezipienten nicht als Fremdkörper empfunden werden.
3. Initiierung eines Dialogs zwischen dem Informationssicherheitsmanagement und den Kollegen, die bisher passiv die Regeln anwenden mussten. Methodisch fundierte und erprobte Ansätze sollen die Basis für die Maßnahmen bilden.
4. Es sollen möglichst innovative werbliche Möglichkeiten ausgeschöpft werden, um den Erfolg zu gewährleisten. Das Programm soll für das Qualitätsmanagement Ansätze zur Ermittlung von Kennzahlen und Erfolgsquoten beinhalten.

2 Grundlagen

Im vorliegenden Abschnitt wird die Fragestellung beleuchtet, welche Faktoren bei einem Security-Awareness-Programm zu berücksichtigen sind. Darauf aufbauend wird in Abschnitt 3 ein Framework entwickelt, das als Grundlage für Awareness-Programme in einer Vielzahl von Institutionen dienen kann. Die Umsetzbarkeit (Abschnitt 4) und Wirksamkeit (Abschnitt 5) wurden schließlich exemplarisch mithilfe der Umsetzung des Programms in der buw Unternehmensgruppe überprüft.

2.1 Motivation schaffen

Einen Ansatz menschliches Verhalten zu beeinflussen, liefert das Elaboration Likelihood Model (vgl. Abbildung 1) [Pet86]. Es beschreibt die Auswirkungen einer Mitteilung auf den Empfänger hinsichtlich seiner Einstellung gegenüber dem Thema der Mitteilung. Nach diesem Prinzip können und werden Personen am besten Informationen verarbeiten und in neues Verhalten umsetzen, für die sie eine hohe Motivation haben und deshalb eine Aufnahmebereitschaft besitzen [Hal98] [EKR08] [PTZ13].

Personen setzen sich auf der sogenannten ‚zentralen Route' kritisch mit der Information auseinander. Daraus resultiert eine Verhaltensänderung und eine neue änderungsresistente Einstellung. Bei fehlender Motivation werden die Informationen auf der ‚peripheren Route' hingegen nur oberflächlich untersucht und führen, wenn überhaupt, zu einer instabilen Einstellungsänderung. Häufig werden diese ungewollten Informationen sogar als störend empfunden und führen zu einer Abneigung gegenüber dem betreffenden Thema oder sogar gegenüber dem Mitteilenden.

Für die gewünschte Einstellungsänderung wird demzufolge Motivation benötigt. Der Anreiz muss so hoch sein, dass er mögliche Unbequemlichkeiten überwiegt. Bezogen auf Security-Awareness ist deshalb Motivation im Sinne eines Wachrüttelns eine tragende Säule.

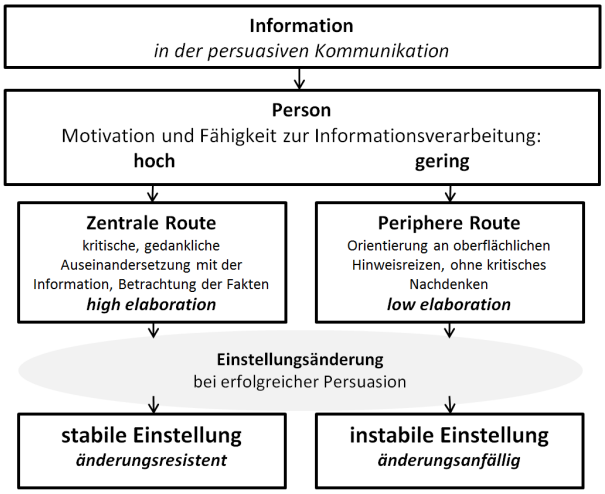


Abbildung 1: Elaboration Likelihood Model [Pet86]

2.2 Wissen vermitteln

Der Sicht von Awareness als wortgetreue Übersetzung, nämlich das Sicherheitsbewusstsein der Mitarbeiter, welches es zu verändern gilt, liegt häufig die Annahme zugrunde, dass die bewusste Einstellung eindeutig mit sicherheitskonformem Verhalten korreliert [Mat09] [mOTG13] [AG13a] [HP09]. Dabei bleibt unberücksichtigt, dass das Bewusstsein über einen Sachverhalt nicht dessen Verständnis impliziert.

Für den Erfolg von Wissensvermittlungen spielen zwei Faktoren eine wesentliche Rolle. Zunächst die Frage, wie gut der Lernstoff vermittelt und verstanden wird. Im Zusammenhang mit Security-Awareness gelingt es in der Regel, die komplexen Themen auf einfache Lerninhalte für die Zielgruppe zu reduzieren. So ist es für den Anwender beispielsweise nicht notwendig etwas über dynamische Schlüssel, das Temporal Key Integrity Protocol (TKIP), Pre-shared keys (PSK) oder Extensible Authentication Protocol (EAP) zu wissen. Zur sicheren Benutzung eines Funknetzes genügt die Information, dass WPA2 gegenüber WEP zu bevorzugen ist [SWA05].

Der zweite und wesentliche Faktor ist der Zeitraum, über den der Lernende diesen Stoff behalten kann. Die Psychologie ist dahingehend gut erforscht und es ist gemeinhin bekannt, wie lange der Mensch neu Gelerntes behält. Der Vorgang des Vergessens kann demnach durch mehrfaches Wiederholen des Lernstoffes abgemindert werden (Überlernen), jede Wiederholung vergrößert das Intervall, nach dem eine erneute Wiederholung notwendig ist [MN90]. Daraus resultiert die Notwendigkeit, ein Awareness-Programm und insbesondere die damit verbundene Wissensvermittlung über einen längeren Zeitraum zu strecken, sodass das Thema in Abständen immer wieder in den Fokus rückt und langfristig Wirkung entfalten kann [cA13].

2.3 Emotionen wecken

Bei der Reduktion von Awareness auf zusätzliche Schulungsmaßnahmen [Rup13] [Inc13] [aG13b], bleibt vielfach unberücksichtigt, dass das neu erworbene Wissen nicht notwendigerweise zu verändertem Verhalten führt.

In der unternehmensinternen Kommunikation konkurriert Informationssicherheit mit einer Vielzahl weiterer, wichtiger Themen wie Mitarbeiterführung, Kundenbetreuung oder Abwicklung des Tagesgeschäfts. Deshalb ist es erforderlich, Sicherheit systematisch im Unternehmen zu vermarkten, damit sie als Top-Thema wahrgenommen wird. Als zusätzliche Herausforderung ruft Sicherheit in der Ausgangssituation eher unangenehme Gefühle hervor. Sie wird mit zusätzlicher Arbeit, Komfortverzicht oder als Freiheitseinschränkung assoziiert.

Im Sinne der Zielerreichung muss Sicherheit positiv besetzt werden und ein akzeptiertes Verhalten sein. Bei der Suche nach möglichst innovativen Kommunikationsformen, fällt vor allem das virale Marketing als geeigneter Kandidat ins Auge. „Virales Marketing beschreibt das gezielte Auslösen von Mundpropaganda zum Zwecke der Vermarktung von Unternehmen und deren Leistungen [Lan09].“ In Anlehnung an den medizinischen Begriff des Virus soll sich dabei die Information über ein Produkt oder eine Dienstleistung epidemisch von Mensch zu Mensch verbreiten, sodass sich das Interesse am beworbenen Produkt und damit dessen Bekanntheitsgrad erhöhen. Diese Vorgehensweise fördert zudem den erwünschten Dialog.

Virales Marketing erzielt durch Emotionalität teilweise beeindruckende Erfolge. Als aktuelles Beispiel für erfolgreiches virales Marketing sei hier ein Projekt des Unternehmens Blendtec vorgestellt: Blendtec, ein Hersteller für Küchengeräte, rief einen Youtube Kanal ins Leben, in dem ein Küchenmixer dazu verwendet wird, Gegenstände zu zerstören. Die Reichweite ist erstaunlich. So wurde die Folge, in der ein i-Pad zerstört wird über 16 Millionen Mal gesehen und fast 50.000 Mal kommentiert [Ble13b]. Laut einer Aussage der Unternehmensführung von Blendtec steigerten sich die Verkäufe nach Einführung der Show um 700% [Ble13a].

Die Vorgehensweise lässt sich auf die Vermarktung von Sicherheit übertragen. Im ersten Schritt erfolgt das sogenannte ‚Seeding‘ (engl. für Impfen oder Aussäen). Mithilfe einer kreativen Idee wird der Inhalt an einer Stelle im Interessentenumfeld platziert. Von dort soll er sich selbständig mit steigender Popularität verbreiten.

3 Framework

Es existieren bereits erprobte Modelle für Security-Awareness-Frameworks. Diese setzen ihren Schwerpunkt z. B. auf den Lebenszyklus eines Mitarbeiters im Unternehmen [Sch13], die starke Einbeziehung der jeweiligen Fachabteilung [KS13] und den Einsatz moderner Medien mit Comics und Web 2.0 Elementen [ED13]. Aus den Grundlagen der vorigen Ausführungen und nach Sichtung bekannter Frameworks wurde für das Programm die Vorgehensweise in Abbildung 2 abgeleitet.

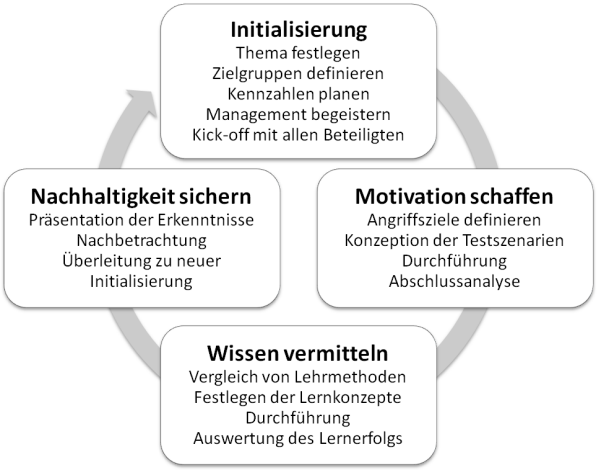


Abbildung 2: Framework für Awareness-Programm [eigene Abbildung]

In einem ersten Initialisierungsprozess gilt es ein geeignetes Thema zu identifizieren und Zielgruppen des Programms festzulegen. Bei der Wahl des Themas sollten die folgenden Faktoren berücksichtigt werden: Wahrscheinlichkeit des Eintretens, Höhe des Schadens, Einflussnahme durch Verhalten der Mitarbeiter. Bereits in dieser Phase ist es unabdingbar die Unternehmensleitung einzubeziehen, da sie nicht nur notwendige Freigaben für die Planung erteilen muss, sondern auch auf die Ausrichtung des Programms strategischen Einfluss nehmen sollte.

In Anlehnung an das gewählte Thema muss ein glaubwürdiges Szenario für die Anwender entwickelt werden. Das Risiko soll erlebbar gemacht werden, in der Regel durch eine Angriffssimulation. Die BSI Studie „Durchführungskonzept für Penetrationstests“ [fSidI03] beschreibt als Vorgehensweise zunächst das Angriffsziel zu definieren und ein Konzept festzulegen. Bei der Konzeptionierung muss an die möglicherweise nötigen Verträge wie z. B. Geheimhaltungsvereinbarung genauso gedacht werden, wie an die Einbeziehung aller wichtigen Beteiligten, wie Betriebsrat oder Datenschutzbeauftragte. Bei der Durchführung sollen Mitarbeiter keinesfalls durch dauernde Wiederholung der Simulationen negative Assoziation erfahren. In der Abschlussanalyse zeigt sich dann, ob die Risiken während der Initialisierung richtig gewählt wurden. Denn nun lässt sich messen, ob von den angenommenen Risiken echte Bedrohungen ausgehen und ob die Anwender durch ihr Verhalten die Gefährdung wirksam abgewendet haben.

Aus der Abschlussanalyse resultieren die Lernziele für die Wissensvermittlung, es sind genau jene Punkte, in denen die Mitarbeiter durch ihr Verhalten ein Risiko für die Informationssicherheit provoziert haben. Für diese Ziele wird eine Lehrmethode gewählt, ein Lehrplan für jede Zielgruppe erstellt und umgesetzt. Dabei sind die positive Vermarktung des Themas und die emotionale Aufladung zu berücksichtigen. Am Ende dieser Phase ist eine weitere Analyse empfehlenswert, um den Wirkungsgrad des Schulungskonzepts zu

bewerten.

Zur Sicherung der Nachhaltigkeit werden die Ergebnisse der Unternehmensleitung präsentiert und für alle Mitarbeiter aufbereitet. Dabei ist die Wahl der Kommunikationsmethode ebenso entscheidend, wie bei der Wissensvermittlung selbst. In einer Analyse sind Verbesserungen für zukünftige Programme zu ermitteln, bevor zu einer neuen Initialisierung übergeleitet werden kann.

4 Praxisbeispiel

Die buw Unternehmensgruppe ist im Bereich Customer Care tätig und unterliegt somit als Auftragsdatenverarbeiter verschiedenen vertraglichen und gesetzlichen Anforderungen nach §11 Bundesdatenschutzgesetz (BDSG). Um die notwendige sicherheitsbewusste Kultur zu etablieren, verfolgt das Unternehmen bereits mehrere Ansätze. Es wurde ein ISMS (InformationsSicherheitsManagementSystem) installiert und nach ISO 27001 auf der Basis von IT-Grundschutz zertifiziert. Auf Grundlage einer unternehmensinternen Sicherheitsrichtlinie „Sensibilisierung und Schulung zur Informationssicherheit“ werden bereits eine Reihe von Sensibilisierungsmaßnahmen durchgeführt. Trotz dieser umfangreichen Vorkehrungen lassen sich Sicherheitsvorfälle auf ungenügendes Wissen oder die Nicht-Einhaltung von Sicherheitsmaßnahmen zurückführen.

4.1 Initialisierung

Basierend auf dem Framework (vgl. Abbildung 2) wurde in der buw Unternehmensgruppe zunächst das Thema festgelegt. Verglichen wurden Risiken im Umgang mit Informationen, wie beispielsweise Wirtschaftsspionage, Sabotage, Datenmanipulation und Datendiebstahl. Als Ergebnis des Vergleichs wurde der Bereich Datendiebstahl als erfolgversprechendstes Thema für das Programm identifiziert. Die Wahrscheinlichkeit eines Verlusts vertraulicher Daten an einen externen Angreifer wäre durch die Vielzahl der Möglichkeiten hoch, der Schaden wäre beträchtlich und korrektes Verhalten der Mitarbeiter wäre eine wirksame Gegenmaßnahme.

Die Zielgruppe wurde auf alle 5.000 Mitarbeiter an 10 Standorten der Unternehmensgruppe festgelegt. Jedoch sollte der Schwerpunkt auf dem deutlich kleineren Anteil der Anwender liegen, die Funktechnologie, einen E-Mail-Account und das Internet verwenden. Dort wurde ein erhöhtes Risiko für Datendiebstahl vermutet. Der zeitliche Rahmen wurde auf 4 Monate angelegt, so dass ausreichend Zeit für die nötige Wissensvermittlung zur Verfügung stand. Zur Kennzahlermittlung wurden die Kriterien Gesamtzahl der Angriffe, Zahl der erfolgreichen Angriffe und die Vertraulichkeitsklasse der entwendeten Informationen betrachtet. Um die Kampagne intern zu präsentieren, konnte auf die Unterstützung des Fachbereichs Marketing zurückgegriffen werden. Daraus resultierte die Gestaltung eines Kampagnenlogos mit einem stilisierten Banditenkopf und das Motto „Bis jetzt ist ja auch nichts passiert. Bis jetzt“. Dem Thema folgend wurden in der Motivationsphase sämtli-

che mit dem Programm zusammenhängende Aktionen mit dem Angriff eines virtuellen Banditen verknüpft. Firmenkulturell waren Benchmarks der Standorte oder Abteilungen gegeneinander bereits etabliert. Im Zuge des Programms wurde diesem bekannten Vorgehen der gemeinsame Feind in Form des Banditen hinzugefügt. Es sollte ein Wettbewerb von Abteilungen sein, mit dem Ziel den Banditen zu besiegen.

4.2 Motivation

Das Szenario bestand aus einem Außentäter, der versuchen sollte, vertrauliche Daten zu stehlen. Dabei wurde die Art eines Blackbox-Tests gewählt. Der extern beauftragte Angreifer erhielt im Vorfeld keine Informationen über das Unternehmen. Nach Abschluss eines Geheimhaltungsvertrags und Rücksprache mit den Datenschutzbeauftragen wurde die Aufgabenstellung formuliert. Das Vorgehen sollte vorsichtig abwägend und eher laienhaft sein. Die interne Erwartungshaltung war deshalb ein Fehlschlagen des Datendiebstahls in allen 5 konkreten Arbeitsaufträgen:

1. Simulierter Telefonangriff: Der Externe rief in unterschiedlichen Unternehmensbereichen an und versuchte sensible Informationen zu erfragen.
2. Unberechtigter Zutritt: Er versuchte sich während der Bürozeiten unberechtigt Zutritt zum Gebäude verschaffen. Dort sammelte er Dokumente und hielt nach ungesperrten Rechnern, unbeaufsichtigten Mobiltelefonen oder mobilen Datenträgern Ausschau.
3. Phishing: Eine E-Mail mit einem Phishing-Link von einer als intern getarnten, jedoch nicht existenten E-Mailadresse und mit deutlichen Auffälligkeiten in der Ansprache wurde an alle Mitarbeiter gesendet. Beim Klick auf den Link, landete der Empfänger auf einem Eingabedialog, optisch ähnlich dem Exchange Webmail. Wenn er dort einen Loginversuch startete, landete er auf einer Sensibilisierungswebseite, die ihn über Phishing aufklärte.
4. Präparierte Speichermedien auf dem Firmengelände: Auf ausgelegten USB-Sticks lag eine Datei ‚[Filmtitel].mp4.exe‘. Wurde sie auf einem Unternehmensrechner ausgeführt, erfasste sie den lokalen Rechnernamen und produzierte ein Warnfenster mit dem Kampagnenmotto.
5. WLAN-Honeypot: Ein präparierter WLAN-Router wurde aufgestellt. Bei Verbindung erschien eine html-Seite mit Logindialog.

Zu jedem Arbeitsauftrag wurde eine konkrete Auswertung der Menge der Ziele und der Menge und Art der gestohlenen Daten ermittelt. Dabei wurde bewusst auf eine personenscharfe Analyse verzichtet. Die Abschlussanalyse ist nicht öffentlich, entgegen der oben erwähnten Erwartungshaltung wurden jedoch bei allen Aufträgen Schwachstellen mit Datenverlust verzeichnet. Das Thema und die Angriffsszenarien wurden demnach richtig gewählt.

Den Abschluss der Motivationsphase bildete eine Präsentation für die Geschäftsführung mit den konkreten Ergebnissen, sowie eine Informationsmail an alle Mitarbeiter. Letztere wurde im Namen des Banditen in Form einer ‚Kriegserklärung' verschickt und war gleichzeitig Auftakt zur Wissensvermittlung.

4.3 Wissen vermitteln

Die ermittelten Schwächen wurden dazu verwendet, passgenau das fehlende Wissen zu identifizieren. Alle Mitarbeiter sollten über die Angriffe informiert und mit Kenntnissen ausgestattet werden, wie die aufgetretenen Fehler in Zukunft konkret vermieden werden können. Ziel war explizit nicht nur die Präsentation vorgefertigter Wissensfragmente, sondern der selbständige Umgang mit Informationssicherheit und die Einbettung der Wissensvermittlung in den Arbeitsalltag.

Zur Lehrmethode wurden unterschiedliche Strategien verglichen. Gegen die Präsenzschulung sprachen die räumliche Distanz der zehn Standorte des Unternehmens, das unterschiedliche Vorwissen der Mitarbeiter, wie auch das nicht individualisierbare Lerntempo. Zudem sollte sich das Programm deutlich von den bereits regelmäßig zu absolvierenden Schulungen zur Informationssicherheit, aber auch fachlichen Themen, abgrenzen. Um möglichst viele oder sogar alle Mitarbeiter des Unternehmens zu erreichen, wären 250 Schulungstermine notwendig gewesen – eine organisatorische Unmöglichkeit im Programmzeitraum.

Der technische Aufwand einer Implementierung über alle Standorte und die Einweisung aller Organisatoren wäre bei einem e-Learning mit immensem Aufwand verbunden und würde hohe Selbstlernkompetenz bei allen Teilnehmern verlangen. Zudem sollten die Lernenden ein Thema, nämlich das Social Engineering, bewerten lernen, bei dem es im Wesentlichen um Kommunikation zwischen Menschen geht. Diesen Inhalt über eine rein technische Plattform zu verteilen, schien unpassend. Zuletzt birgt ein e-Learning ohne Ansprechpartner immer eine erhöhte Gefahr der Missdeutung von Inhalten.

Die Lehrmethode sollte in den Alltag eingebettet sein, zu Aktivität anregen und innovativ die geweckte Neugier aus der vorigen Angriffsituation befriedigen. Die Wahl fiel auf eine Einbettung der Wissensvermittlung in ein Spiel. Verwendete Elemente sollten unter anderem Highscores, Fortschrittsbalken, Ranglisten und Auszeichnungen sein. Datenanalysen zeigen signifikante Verbesserungen in Bereichen wie Benutzermotivation, Lernerfolg, Verhaltensänderung, Lerngeschwindigkeit und Nachhaltigkeit bei deren Verwendung [Cha13]. Bereits 2011 antizipierte eine Gartner Studie, dass ‚Gamifizierung' ein signifikanter Trend sei, den bis 2014 über 70% der Top-2.000-Unternehmen eingebunden haben werden [Por12].

Eine Befragung in einem Unternehmen, das ein ‚Alternate Reality Game', unter anderem zur Spamabwehr absolviert hatte, ergab, dass nach dem Programm 95% der Befragten eine Spam E-Mail korrekt erkennen konnten und alle Befragten an einem weiteren Programm teilnehmen wollten [elT10]. Ein Alternate Reality Game zeichnet sich dadurch aus, dass es die Grenzen zwischen Spiel und Wirklichkeit bewusst verwischt. Die Mitspieler betreten über ein sogenanntes ‚Rabbit Hole', ein unerwartetes Ereignis im Alltag, das Spiel. Die Spielinhalte werden über unterschiedliche Medien in den Alltag verlagert. Diese Methode

schien nach Bewertung der Anforderungen passend für die Wissensvermittlung in der buw Unternehmensgruppe geeignet zu sein. Das Spiel sollte sich nahtlos in den Arbeitsalltag einfügen und die Teilnehmer mit unterschiedlichen Medien für das Thema interessieren. Der Ablauf wurde wie folgt festgelegt.

Level 1

Das Seeding, also die kreative Idee, die ausgesät wurde, war zugleich der Zugang zum Alternate Reality Game: das Rabbit Hole. Per Post bekam jede Abteilung eine Holzkiste mit Geheimfach. Im Geheimfach befand sich ein Logical-Rätsel. Zur Lösung waren Wissen über richtige Verhaltensweisen im Unternehmen, wie auch logisches Denken gefordert. Aus der Lösung des Rätsels ergab sich ein Link zu einer Webseite im Intranet. Diese Kampagnenseite wurde während des gesamten Spiels der Anlaufpunkt für alle Mitspieler. Auf der Webseite befanden sich eine Übersicht aller Standorte und Abteilungen mit Anzahl erfolgreicher Spieler, Link zu einem Wissensquiz, ein Counter bereits gelöster Quizzes, ein Infoticker mit aktuellen Neuigkeiten und ein Chat zur Kontaktaufnahme.

Level 2

Auf der Kampagnenseite erschien ein Hinweis im Ticker, dass Level 2 eröffnet sei. Klicken konnte man dieses Mal allerdings nichts. Zeitgleich erschien der turnusmäßige Datenschutz-Newsletter mit einem unauffälligen Morsecode. Der Code führte auf das zweite Quiz zum Thema mobile Datenträger. Mitarbeiter, die dieses Training absolviert haben, füllten damit den zweiten Counter.

Level 3

Nach einer angemessenen Lernzeit erfolgte auf der Kampagnenseite der Hinweis auf das nächste und letzte Level und ein Hinweis auf die im Unternehmen vorhandene Clean-Desk-Policy. In der darauffolgenden Woche wurden Karten auf den Schreibtischen ausgelegt, auf denen sich neben dem Dank für den aufgeräumten Schreibtisch auch je eine Rechenaufgabe befand. Die Lösung aller Aufgaben ergab eine interne Telefonnummer, unter der der Bandit zum letzten Test mit dem Thema WLAN aufforderte.

4.4 Nachhaltigkeit sichern

Alle Analysen, also sowohl die Angriffsergebnisse, wie auch der Verlauf des Schulungsspiels, wurden zu einer Abschlusspräsentation für die Geschäftsführung zusammengefasst. Die Angriffszenarien und die Merksätze wurden außerdem in Form eines Posters und einer Intranetseite für die Mitarbeiter dauerhaft zugänglich gemacht. Zudem wurde das Programm mit mehreren Beiträgen im Firmenblog und einem Bericht in der Firmenzeitung für Mitarbeiter und Kunden besonders gewürdigt. Auf der Intranetseite fand eine Abschlussbefragung statt, um Reichweite und Eindrücke der Teilnehmer messbar zu machen.

5 Bewertung

Das konzipierte Programm war ein Erfolg. Es konnte im Rahmen des vorgegebenen Frameworks umgesetzt werden. Die Beschäftigung mit Informationssicherheit in spielerischer Form wurde positiv wahrgenommen. In der Befragung gaben 47% an, dass sie gern mitgespielt haben und bei einer Wiederholung erneut mitspielen würden. Im Vergleich dazu äußerten sich nur 3% kritisch und würden bei einer Wiederholung nicht mehr teilnehmen. Es ist zu eruieren, was für Gründe diese Mitspieler für ihre Entscheidung hatten.

Unmittelbar nach Spielbeginn bekam der Fachbereich für Informationssicherheit mehrfach Feedback und gezielte Fragen nach Lösungshinweisen. Exemplarisch sei hier das Zitat einer Mitarbeiter-E-Mail wiedergegeben: „Der sportliche Ehrgeiz ist geweckt! Auch wenn wir am Ende vielleicht nicht die Nase vorn haben sollten – wir haben den Kampf angenommen! Wir rühren jedenfalls mächtig die Werbetrommel und verteilen Quizhefte und vergrößerte Lösungsmatrizen."

Auch konkrete Sicherheitsüberlegungen wurden geschildert und mit den hausinternen Ansprechpartnern diskutiert. Die Mitspieler kamen aktiv auf die Idee Hilfsmittel einzusetzen und beschäftigten sich mit großem Engagement mit der Informationssicherheit. Im Verlauf des Spiels sank der Anteil der falsch beantworteten Quizfragen, ein Hinweis auf den Erfolg der Lernphase. Zudem wurden aus der Angriffsphase wertvolle Erkenntnisse über Prozesschwächen und Risiken gewonnen.

Viralität

Das Spiel wurde mit 77 Holzkisten initiiert. Am Ende des ersten Levels waren bereits über 700 Quiz gelöst. Neben dem persönlichen Austausch mit Kollegen fand die Verbreitung auch Wege über Intranet, Telefonate und E-Mails. Das Spiel wurde zum Thema im Unternehmen. Es bekam einen ruckartig ansteigenden Zulauf, als der erste Beitrag im Firmenblog dazu erschien. Weitere, bisher unbeteiligte Mitarbeiter beteiligten sich. Bis zum Ende des zweiten Levels wurden mehr als 3.000 Quizfragen beantwortet. Insgesamt wurden 13.150 Quizfragen im Verlauf des gesamten Zeitraums beantwortet, davon 6.375 korrekt.

Verbesserungsansätze

Es ist nicht vollständig gelungen, die sehr große Zielgruppe mit den für sie passenden Inhalten zu versorgen. Bei einer Wiederholung sollte das Framework deshalb auf eine kleinere Zielgruppe angewendet werden. Der erste Schritt im Spiel war aus Gründen der Umsetzbarkeit an die Abteilungsleitung und nicht an jeden einzelnen Mitarbeiter adressiert. In Ausnahmefällen ist deshalb die Spieleinladung nicht über die Leitungsebene bis in die Abteilung vorgedrungen. Diese Erkenntnis spiegelt sich auch in der Befragung wieder, bei der 36% angaben, dass sie von dem gesamten Spielverlauf nichts mitbekommen hätten. Bei einer Wiederholung dieses Programms, scheint es empfehlenswert diese Herangehensweise zu überarbeiten. So könnte eine noch breitere Streuung erreicht werden.

Literatur

[AG13a] Aconsite AG. *Wie Sie mit aufgeweckten Mitarbeitern Datenschutzskandale vermeiden*. Website: http://www.aconsite.de/CMS/Security-Awareness.html, Abruf: 5.12.2013, 2013.

[aG13b] ausecus GmbH. *Schulungen und Workshops*. Webseite: http://www.ausecus.com/de/industrial-it-security/awareness-workshops, Abruf: 5.12.2013, 2013.

[Ble13a] Blendtec. *Video: BlendTec CEO Says Sales up 700% Since Launching: Will it Blend*. Website: http://www.youtube.com/watch?v=u6t92m1gwTY, Abruf: 5.12.2013, 2013.

[Ble13b] Blendtec. *Video: Will It Blend? – iPad*. Website: http://www.youtube.com/watch?v=lAl28d6tbko, Abruf: 5.12.2013, 2013.

[cA13] Hvs consulting AG. *Security Awareness Kampagnen*. Website: http://www.hvs-consulting.de/security-awareness-kampagnen.aspx, Abruf: 5.12.2013, 2013.

[Cha13] Bill Chamberlin. *Gamification: A 2013 HorizonWatching Trend Report. HorizonWatching and IBM*. Website: http://de.slideshare.net/HorizonWatching/gamification-a-horizon-watching-trend-report-05feb2013, Abruf: 5.12.2013, 2013.

[DGHA12] European Commission Directorate-General Home Affairs. *Cyber Security Report*. Website: http://ec.europa.eu/public_opinion/archives/ebs/ebs_404_en.pdf, Abruf: 5.12.2013, 2012.

[ED13] Andreas Exter und Matthias Drodt. *CSI:DB – Die IT Security-Awareness-Kampagne der Deutschen Bahn*. Datenschutz und Datensicherheit 5.2013, 2013.

[EKR08] Martin Eisend und Franziska Küster-Rohde. Soziale Netzwerke im Internet — Marketingkommunikation für morgen. *Marketing Review St. Gallen*, 25(5):12–15, 2008.

[elT10] enspire learning Texas. *Whitepaper This Is Not A Game: Using Alternate Reality Games in Corporate Training*. Website: http://www.enspire.com/wp-content/uploads/2010/09/White-Paper-Using-Alternate-Reality-Games-in-Corporate-Training.pdf, Abruf: 5.12.2013, 2010.

[Fon12] John Fontana. *Employees would sell work password for 5 pound*. Webseite: http://www.telegraph.co.uk/technology/internet/9189236/Employees-would-sell-work-password-for-5.html, Abruf: 5.12.2013, 2012.

[fSidI03] Bundesamt für Sicherheit in der Informationstechnik. *Durchführungskonzept für Penetrationstests*. Website: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest_pdf.pdf, Abruf: 13.02.2014, 2003.

[Hal98] Gregor Halff. Ergebnis aus der Strukturierung der Involvement-Forschung: das 'Elaboration Likelihood Model' ('ELM'). In *Die Malaise der Medienwirkungsforschung: Transklassische Wirkungen und klassische Forschung*, Jgg. 28 of *Studien zur Kommunikationswissenschaft*, Seiten 175–195. VS Verlag für Sozialwissenschaften, 1998.

[HP09] Michael Helisch und Dietmar Pokoyski. *Security Awareness. Neue Wege zur erfolgreichen Mitarbeiter-Sensibilisierung*, Seite 10. kes vieweg und teubner, 2009.

[Inc13] McAfee Inc. *Security Awareness Program Development and Training*. Webseite: http://www.mcafee.com/de/services/strategic-consulting/program-development/security-awareness-program-development-and-training.aspx, Abruf: 5.12.2013, 2013.

[KS13] Robert Kaltenböck und Sabine Schuster. *Awareness für Informationssicherheit und Datenschutz in der Sparkassen-Finanzgruppe. Mit Bildsprache Mitarbeiter sensibilisieren.* Datenschutz und Datensicherheit 5.2013, 2013.

[Lan09] Sascha Langner. *Viral Marketing. Wie Sie Mundpropaganda gezielt auslösen und Gewinn bringend nutzen. 3. erweiterte Auflage*. Wiesbaden: Gabler, GWV Fachverlage GmbH, 2009.

[Mat09] Jochen Matzer. *Security Awareness. Neue Wege zur erfolgreichen Mitarbeiter-Sensibilisierung*. kes vieweg und teubner, 2009.

[MN90] Christian Michel und Felix Novak. *Kleines psychologisches Wörterbuch. Erweiterte und aktualisierte Neuausgabe. ISBN 3-451-08690-5*. Herder, Freiburg (Breisgau), 1990.

[mOTG13] mybreev Online Trainings GmbH. *awareness training ändert die Verhaltensmuster der Mitarbeiter*. Website: http://www.mybreev.com/de/news/awareness-training-andert-die-verhaltensmuster-der-mitarbeiter.html, Abruf:5.12.2013, 2013.

[Pet86] Cacioppo Petty. *The Elaboration Likelihood Model Of Persuasion. In: Advances in experimental social psychology*, Seiten 123–205. New York: Academic Press, 1986.

[Por12] Gartner Portals. *Gartner Predicts Over 70 Percent of Global 2000 Organisations Will Have at Least One Gamified Application by 2014. Content and Collaboration Summit, March 12-14, 2012 in Orlando, Florida*. Website: http://www.gartner.com/newsroom/id/1844115, Abruf: 5.12.2013, 2012.

[PTZ13] Polyxeni (Jenny) Palla, Rodoula H. Tsiotsou und Yorgos C. Zotos. Is Website Interactivity Always Beneficial? An Elaboration Likelihood Model Approach. In Sara Rosengren, Micael Dahlén und Shintaro Okazaki, Hrsg., *Advances in Advertising Research (Vol. IV)*, EAA Series, Seiten 131–145. Springer Fachmedien Wiesbaden, 2013.

[Ros12] Peter Rossbach. *Der Mitarbeiter als Komponente der Informationssicherheit*. Frankfurt School of Finance u. Management, 2012.

[Rup13] Benjamin Rupp. *In Frankfurter Unternehmen sorgt AirITSystems für mehr Sicherheit*. Webseite: http://www.airitsystems.de/standorte/frankfurt/security-awareness-frankfurt/, Abruf: 5.12.2013, 2013.

[Sch13] Klaus Schimmer. *Ein Poster ist zu wenig! Das Trainings- und Awareness Framework der SAP*. Datenschutz und Datensicherheit 5.2013, 2013.

[SWA05] Dorothy Stanley, Jesse Walker und Bernard Aboba. *Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs*. RFC 4017 (Informational), http://www.ietf.org/rfc/rfc4017.txt, 2005.

[ZAO13] Kaspersky Lab ZAO. *The evolution of Phishing attacks: 2011-2013*. Website: http://media.kaspersky.com/pdf/Kaspersky_Lab_KSN_report_The_Evolution_of_Phishing_Attacks_2011-2013.pdf, Abruf: 5.12.2013, 2013.

Hardware Efficient Authentication based on Random Selection

Frederik Armknecht, Matthias Hamann, Matthias Krause

University of Mannheim
{armknecht,hamann,krause}@uni-mannheim.de

Abstract: Lightweight authentication protocols based on random selection were introduced as an alternative design paradigm besides the usage of lightweight block ciphers and the principle of adding based noise. However a comparatively large key length and the use of involved operations made a hardware-efficient implementation a challenging task. In this work we introduce the $(n, k, L)^{80}$-protocol, a variant of linear authentication protocols which overcomes these problems, and analyze its security against all currently known, relevant passive and active attacks. Moreover, we present an implementation of our protocol for FPGAs and ASICs using Verilog and discuss its efficiency w.r.t. generally accepted costs metrics. The respective numbers show that the $(n, k, L)^{80}$-protocol is a viable alternative to existing solutions and is, for example, well suited for the implementation on passive RFID tags.

1 Introduction

1.1 Lightweight Authentication Protocols

Devices of extremely limited computational power like (passive) radio frequency identification (RFID) tags are used in practice to a rapidly growing extent, a trend commonly referred to as ubiquitous computing. One of the major use-cases for such pervasive devices are authentication solutions, e.g., access control for buildings or cars, electronic passports or even human-implantable chips providing sensitive medical information about a person. Consequently, the search for lightweight authentication protocols became an important topic in cryptography during the last years with high relevance for academia and industry.

Today, one can distinguish three main approaches for constructing lightweight authentication protocols:

1. protocols which use lightweight block ciphers like PRESENT [BKL$^+$07], KATAN and KTANTAN [DDK09] as basic cryptographic operations,
2. protocols which employ the well-researched principle of adding biased noise to a secret linear function,
3. protocols which are based on the principle of random selection, being the most recent of all three paradigms.

Concerning approach 1.), it has to be stated that very convincing proposals for lightweight block ciphers as PRESENT, KATAN and KTANTAN do exist which have been analyzed in a large number of papers (e.g., [BKL$^+$07], [DDK09], [KMNP11], [Å11]). However such protocols are less flexible with respect to scalability than other approaches.

Concerning approach 2.), the security of these kinds of (HB-type) protocols w.r.t passive attackers can be reduced to the widely accepted hardness of the learning parity in the presence of noise (LPN) assumption. A severe drawback of these protocols is that presumably secure parameter combinations imply large amounts of transmitted data. Together with the small available bandwidth in RFID communication, this may add up to authentication times that are unacceptable for many applications. A further major problem is that almost all variants, i.e., [JW05, GRS08, BC08] were broken by active man-in-the-middle (MITM) attacks, e.g., see [GRS05, OOV08, FS09]. The only exception we are aware of are the proposals in [KPC$^+$11] and [HKL$^+$12], which are both based on modified variants of the original LPN problem. However, even the latter (more efficient) one of these is "targeting lightweight tags that are equipped with (small) CPUs" [HKL$^+$12] due to the fact that the protocol's computational complexity would violate common timing constraints on cheaper, less powerful hardware (which is targeted in this work).

Approach 3.), i.e., the principle of random selection, implies that the secret key K consists of a small collection of L linear mappings. The prover (e.g., an RFID tag) computes responses to challenges $a \in GF(2)^n$, $n \in \mathbb{N}$, by choosing one of these functions $f \in K$ and replying with $f(a')$, where a' depends on a in a way we are going to specify concretely as part of section 2. The first protocols of this kind were the CKK-protocols given in [CKo08]. Further protocols based on the principle of random selection include the F_f-protocols in [BKM$^+$09] and the Linear Protocols in [KS09]. The most important and still unbroken suggestion of the latter type is the $(n, k, L)^{++}$-protocol also given in [KS09]. It has been proved that the $(n, k, L)^{++}$-protocol is resistant w.r.t. to a wide family of active MITM attacks. Moreover, the security of $(n, k, L)^{++}$-protocols can be reduced to the complexity of the problem of learning unions of linear subspaces (LULS problem). In analogy to HB-type protocols, the security of Linear Protocols is thus based on the assumption that it is impossible to solve the LULS problem in a more efficient way. In [KH11] the best approach known so far for solving the LULS problem has been given. Its effort is dominated by the cost of inverting a matrix of size n^L.

1.2 Our Contribution

In previous works about $(n, k, L)^{++}$-protocols, two problems w.r.t. efficiency were left open for future research and prevented this type of protocol from being practically used so far: Firstly, the large key length resulting from the need to specify the afore-mentioned set of secret linear functions. Secondly, certain operations deemed necessary in order to achieve MITM-security were still too demanding in hardware.

The $(n, k, L)^{80}$-protocol introduced in this paper aims at solving both of these problems. In particular, we are able to reduce the key length to a feasible size of 80 bits and show

that the security reductions presented in [KS09] and [KH11] still apply to a large extent. Moreover, all operations used in the $(n, k, L)^{80}$-protocol can be realized efficiently in hardware. In order to show this, we created an actual implementation for FPGAs and ASICs using the hardware description language Verilog and present the corresponding efficiency metrics, which indicate that the suggested protocol is a viable alternative to prevalent block cipher based constructions.

Table 1 in appendix A provides an overview of our results, which are explained in further detail as part of section 4. Most notably, the suggested protocol can be safely realized at costs below 1,300 GEs (Gate Equivalents) without succumbing to the attacks described in section 3 (or having been broken by other means). The in-depth description of our design in subsection 2.2 will explain why choosing smaller parameters reduces the area costs as well as the total number of needed clock cycles (despite an increase in rounds) while leaving the communication complexity unchanged.

2 A Proposal for a Hardware Efficient Linear Protocol

2.1 The $(n, k, L)^{++}$-Protocol

We first recall the definition of $(n, k, L)^{++}$-protocols as it was suggested in [KS09]. In the original specification, it is a one-round challenge-response authentication protocol, whose symmetric key consists of a small number L of injective linear functions $F_1, \ldots, F_L : \{0,1\}^n \longrightarrow \{0,1\}^{n+k}$. Based on theoretical considerations as well as experimental results (see also section 3), the following parameter sizes were suggested: $n = 128, k = 32, L = 8$. Figure 1 in appendix B depicts an instance of the $(n, k, L)^{++}$-protocol for a verifier Alice (RFID reader) and a prover Bob (RFID tag).

The authentication process is initiated by Alice who chooses uniformly and at random a challenge $a \in_U GF(2)^{\frac{n}{2}}$, $a \neq \mathbf{0}$, and sends it to the prover. Likewise, the prover chooses a random nonce $b \in_U GF(2)^{\frac{n}{2}}$, $b \neq \mathbf{0}$, of the same length, randomly picks one of the L secret linear functions $F_1, \ldots, F_L$, and responds $w = F_l\left(f\left(a, b\right)\right)$. The non-linear bijective connection function $f : GF\left(2^{\frac{n}{2}}\right)^* \times GF\left(2^{\frac{n}{2}}\right)^* \longrightarrow GF\left(2^{\frac{n}{2}}\right)^* \times GF\left(2^{\frac{n}{2}}\right)^*$, where $GF\left(2^{\frac{n}{2}}\right)^*$ denotes $GF\left(2^{\frac{n}{2}}\right) \setminus \{0\}$, is defined by $f\left(a, b\right) = \left(ab, ab^3\right)$. It is included for thwarting a certain class of man-in-the-middle (MITM) attacks (see subsection 3.3). In order to verify the prover's response, the reader Alice first checks whether w belongs to one of the L n-dimenensional subspaces $V_1, \ldots, V_L$ of $GF(2)^{n+k}$, which are the images of the corresponding injective linear functions $F_1, \ldots, F_L$. Given that $w \in V_l$ holds, Alice subsequently computes $(\tilde{a}, \tilde{b}) = f^{-1}\left(F_l^{-1}(w)\right)$. Finally, if $\tilde{a}$ equals the initial challenge a, Alice will accept the prover's valid response.

The security of this protocol with respect to passive attackers can be reduced to the hardness of the Learning Unions of Linear Subspaces (LULS) problem. In a nutshell, the LULS problem is to learn specifications of the subspaces $V_1, \ldots, V_L$ from independently and uniformly chosen random samples from $\bigcup_{l=1}^{L} V_l$. The complexity of this problem was studied in [KS09] for $L = 2$ and for general L in [KH11]. The best solving algorithm

known so far is an algebraic attack approach which takes time proportional to the time needed to invert a regular matrix of size $O(n^L)$.

A crucial open problem of the original $(n,k,L)^{++}$-protocol was the apparently large key length. As each of the L secret n-dimensional, injective linear functions $F_1,\ldots,F_L : \{0,1\}^n \longrightarrow \{0,1\}^{n+k}$ can be expressed as a distinct $((n+k)\times n)$-matrix over $GF(2)$, in total $((n+k)\cdot n)\cdot L$ bits would need to be stored permanently, which is clearly infeasible for suggested parameter sizes like $n=128$, $k=32$ and $L=8$.

Moreover, for such parameters sizes, even the "simple" non-linear connection function $f(a,b)=\left(ab, ab^3\right)$ induces a big computational overhead in the form of several multiplications over $GF\left(2^{\frac{n}{2}}\right)^*$. Analogously, lookup tables, e.g., in order to efficiently compute b^3 would become very expensive in terms of space.

2.2 The $(n,k,L)^{80}$-Protocol

In this section we introduce the new $(n,k,L)^{80}$-protocol to overcome the two problems mentioned above. In short, the basic ideas are summarized as follows:

- To shorten the key length, the linear functions are no longer randomly sampled and stored but are computed from a smaller seed.
- To lower the effort of the connection function, we replace it by several subfunctions which compute the same functionality but on a smaller domain.

One consequence of these modifications is that the protocol needs to be executed several times. In the following, we first explain all modifications in further detail and provide a description of the overall protocol afterwards.

Shortening the Key Length. The basic idea is to take a keystream generator G that uses a seed of length $m+M$ to (pseudorandomly) generate the $((n+k)\cdot n)\cdot L$ key bits characterizing the secret linear functions $F_1,\ldots,F_L$. In particular, we suppose that $L=2^M$ for a small $M\in\mathbb{N}$ (e.g., $M=4$) and represent each index l, $1\le l\le L$, as an M-bit string $\tilde{l}$. Hence, given a secret symmetric session key $\kappa=(\kappa_1,\ldots,\kappa_m)$, the entries of the matrix corresponding to F_l are certain bits from the key stream produced by G on $(\kappa,\tilde{l})$. Striving for a lightweight construction, it might be tempting to employ a single linear feedback shift register (LFSR) as a simple bitstream generator G. However, we show in subsection 3.2 that allowing the matrices of $F_1,\ldots,F_L$ to be generated by a keystream of small linear complexity opens the door to an algebraic attack which is much more efficient than the afore-mentioned algorithm from [KH11].

Splitting the Connection Function. Another open problem was to reduce the cost introduced by the so-called connection function $f : GF\left(2^{\frac{n}{2}}\right)^* \times GF\left(2^{\frac{n}{2}}\right)^* \longrightarrow GF\left(2^{\frac{n}{2}}\right)^* \times GF\left(2^{\frac{n}{2}}\right)^*$, which is applied to the random values $a,b\in GF\left(2^{\frac{n}{2}}\right)$, $a,b\neq\mathbf{0}$, before they

are fed into one of the L secret linear functions $F_1, \ldots, F_L$. Instead of using $f(a,b) = \left(ab, ab^3\right)$ as a connection function (and thus multiplications over $GF\left(2^{\frac{n}{2}}\right)$), in the new $(n,k,L)^{80}$-protocol, we compute $f(a,b) = \left(\left(a_1b_1, a_1b_1^3\right), \ldots, \left(a_{n/8}b_{n/8}, a_{n/8}b_{n/8}^3\right)\right)$ where $a_i, b_i \in GF\left(2^4\right)$, $a_i, b_i \neq 0$, are obtained by splitting a and b into blocks of 4 bits, respectively. The practical security implications of this modification, which reduces the number of valid challenge-nonce pairs (a,b) from $\left(2^{n/2}-1\right)^2$ to $\left(2^4-1\right)^{n/4}$, are mainly confined to the active attack discussed in section 3.3.

Further Modification. On contrast to the (practically infeasible) $(n,k,L)^{++}$-protocol, it is necessary to run the $(n,k,L)^{80}$-protocol multiple times in order to obtain sufficient resistance w.r.t. certain MIMT attacks. The reason for this is twofold: Firstly, for efficiency reasons, the implementation outlined in section 4 uses challenge-nonce tuples of length $n = 64$ or smaller as compared to $n = 128$ suggested for $(n,k,L)^{++}$. Secondly, one has to compensate for the afore-mentioned decrease of valid inputs (a,b) resulting from splitting up a and b into blocks of size 4 bits each as part of the modified connection function. In subsection 3.3 we show that these modifications lead to an upper bound of $2^{-n/4}$ (e.g., 2^{16} for $n = 64$) for the success probability of a certain MITM attacker to convince an honest verifier to accept an illegitimate response. As this success probability is too large for practical applications, one has to run the protocol at least two times, which would, e.g., lead to an upper bound of $2^{-n/2}$ due to the fact that the rounds can be considered independent w.r.t. the details of this type of attack. As a final modification to the original $(n,k,L)^{++}$-protocol, we introduce a (publicly known) bit-wise permutation σ to the n-bit result of $f(a,b)$. Note that in terms of hardware efficiency, such a bit-wise permutation comes at practically no cost as it is realized simply through wires and does not involve any additional gates.

Protocol Description. The $(n,k,L)^{80}$-protocol proceeds according to the scheme described in subsection 2.1. Again, the process is initiated by the verifier Alice, who chooses some $a \in_U GF(2)^{\frac{n}{2}}$ uniformly and at random and sends it to the prover Bob. Bob then also randomly chooses some $b \in_U GF(2)^{\frac{n}{2}}$ and l, $1 \leq l \leq L$, and answers with

$$w = F_l\left(\sigma\left(f(a,b)\right)\right) = F_l\left(\sigma\left(\left(a_1b_1, a_1b_1^3\right), \ldots, \left(a_{n/8}b_{n/8}, a_{n/8}b_{n/8}^3\right)\right)\right)$$

as described previously. Remember that, in order to allow for inverting $f(a,b)$ as part of the verification, only challenges a and nonces b satisfying $a_i, b_i \neq 0$ for $i \in \left\{1..\frac{n}{8}\right\}$ are allowed by the protocol. The verification step of Alice is exactly the same as for the $(n,k,L)^{++}$-protocol, see subsection 2.1. Please note that while a might be known to an adversary eavesdropping on the communication between Alice and Bob, b is kept strictly secret by the prover and is only used to compute $\sigma\left(f(a,b)\right)$. We will denote $\sigma\left(f(a,b)\right) = (x_0, \ldots, x_{n-1}) = x$ during the following steps. Let us now consider an example where $L = 16$ (public known) and the prover Bob randomly and secretly chooses $l = 6$. Consequently, Bob would have to compute $F_6(x)$ and send the resulting $(n+k)$-bit string to the verifier Alice. As outlined previously, in order to achieve a feasible key

length for the $(n, k, L)^{80}$-protocol, we deploy a keystream generator G with 80-bit initial state to specify the secret linear functions $F_1, \ldots, F_L$. The common secret shared between the verifier Alice and the prover Bob comprises of $80-log_2(L) = 76$ bits $\kappa = (\kappa_1 \ldots \kappa_{76})$. In order to derive a specification of F_6 (needed to compute $F_6(x)$), Bob concatenates the bit string $\tilde{l} = 0101$, the binary representation of the 6th function, and the 76 common secret bits, yielding the corresponding 80-bit seed $\kappa||\tilde{l}$ of G. The resulting keystream $z = z_0 z_1 z_2 \ldots$ enters the computation of the $n + k$ bits $y_0 \ldots y_{n+k-1}$ of $F_l(x)$, i.e. the final authentication token, as follows:

$$\begin{aligned} y_0 &= z_0 \cdot x_0 \oplus \ldots \oplus z_{n-1} \oplus x_{n-1}, \\ &\ldots \\ y_{n+k-1} &= z_{(n+k-1)\cdot n} \cdot x_0 \oplus \ldots \oplus z_{(n+k-1)\cdot n+(n-1)} \oplus x_{n-1}. \end{aligned} \tag{1}$$

It should be noted that in the course of computing, e.g., y_0, only a one-bit-wide register is needed in hardware, i.e., firstly, $z_0 \cdot x_0$ is computed and stored, then $z_1 \cdot x_1$ is XORed, and so on, until $z_{n-1} \cdot x_{n-1}$ has been added and y_0 is finally ready to be transmitted to the verifier. This is an important property as registers are especially costly in terms of area and power consumption, so that their use should be restricted to an absolute minimum when designing lightweight cryptographic protocols. While we trade in clock cycles for a reduction of area (and thus power) in several parts of the $(n, k, L)^{80}$-protocol (see, e.g., the above paragraph), the hardware implementation outlined in section 4 also contains measures to reduce the time complexity where possible. Most notably, the block-wise evaluation of $f(a, b)$ can be performed in parallel to the initialization phase of the generator G without inducing any additional hardware cost. This allows to start computing the first token bit y_0 instantly once G (e.g., a self-shrinking generator based on an LFSR) is ready.

3 Security Analysis

3.1 General Remarks

In this section we analyze the security of the $(n, k, L)^{80}$-protocol, which is, as pointed out previously, in fact a variant of the $(n, k, L)^{++}$ authentication protocols where some modifications have been made for improving the hardware efficiency. In a nutshell, these modifications are (cf. Sec. 2):

- The linear functions F_l are not randomly chosen but generated from a common seed, using a bitstream generator G.
- The connection function has been broken down into several subfunctions which all realize in principle the same function, but restricted to a smaller domain.

Consequently, we investigate if and to what extent these modifications impact the security of the $(n, k, L)^{80}$-protocol in comparison to the security of the $(n, k, L)^{++}$-protocol.

With respect to the latter, we want to point out that the best attacks known so far against $(n, k, L)^{++}$-type protocols are a passive algebraic attack (cf. [KH11]) and an active MITM attack (cf. [KS09]).

3.2 Impact of Using a Generator G

In this subsection we investigate the security impact if the linear functions F_l are not randomly chosen but derived from a bitstream generated by a generator G. To this end, in appendix C, we demonstrate that if G is *weak* (more precisely, where the generated bitstream exhibits a small linear complexity) the whole protocol becomes vulnerable to the passive algebraic attack from [KH11]. This shows the necessity for stronger generators. In fact, we will argue now, using a standard hybrid argument, that using G does not imply any significant change in security if G produces a pseudorandom bitstream (as it is commonly expected from secure keystream generators).

Security Reduction for Pseudorandom-Bit-Generators G. Next we consider the case that G is instantiated by a bitstream generator which produces a bitstream (z_i) of pseudorandom bits given a seed $\alpha \in GF(2)^\ell$. More precisely, let $q = ((n+k) \cdot n) \cdot L$ be the number of bits that characterize the secret linear functions $F_1, \ldots, F_L$. For simplicity, we assume that the first q outputs of G eventually define the linear functions. Now, let G be a (q, t, ε)-secure pseudorandom bit generator and let $\vec{z} = (z_0, \ldots, z_{q-1})$ be a bitstring of length q. This means that for any algorithm D which accepts q bits input and which runs in time t, it holds

$$\begin{aligned} |\Pr\left(1 \leftarrow D(\vec{z}) | \vec{z} \leftarrow G(\alpha), \alpha \in_U GF(2)^\ell\right) \\ -\Pr\left(1 \leftarrow D(\vec{z}) | \vec{z} \in_U GF(2)^q\right)| \leq \varepsilon. \end{aligned} \tag{2}$$

Using a standard argument, one can show that the success probability of any attacker A against the protocol using G deviates at most by ε from the success probability if the linear functions are characterized by uniformly and independently sampled bits. More precisely, let A denote any attacker against the $(n, k, L)^{80}$-protocol which runs in time t at most. We define a corresponding security experiment Exp_A which is equal to 1 if A has been successful. Moreover, we consider two games. In $\mathsf{Game\ 0}$, the linear functions F_l have been determined by the output of G based on a secret seed, while in $\mathsf{Game\ 1}$, they are characterized by independently and uniformly sampled bits. The latter corresponds to a situation where the linear functions are randomly chosen, as suggested in the context of $(n, k, L)^{++}$-protocols. It follows from (2) that

$$|\Pr\left(\mathsf{Exp}_A = 1 | \mathsf{Game\ 0}\right) - \Pr\left(\mathsf{Exp}_A = 1 | \mathsf{Game\ 1}\right)| \leq \varepsilon. \tag{3}$$

Otherwise A could be used directly as a distinguisher for telling apart random bits from outputs of G, hence violating (2). Summing up, if G is a (q, t, ε)-secure pseudorandom bit generator for a sufficiently small value ε, we can practically restrict to the case that the linear functions are randomly chosen. In particular, using generator G yields at most

a negligible difference w.r.t. the security against the passive algebraic attack (cf. [KH11]) and the active MITM attack (cf. [KS09]) in comparison to the $(n, k, L)^{++}$-protocols.

Of course the parameters need to be chosen carefully. The choice of n provides infeasibility of exhaustive key search, while the choice of L has to guarantee resistance against the algebraic attack approach summarized in subsection 3.2. The choice of the parameter k, in turn, must ensure that the following probabilities are negligibly small:

1.) the probability that one of the functions F_l, $l = 1, \ldots, L$, is not injective,

2.) the probability that a random vector $w \in GF(2)^{n+k}$ falls into $\bigcup_{l=1}^{L} V_l$,

3.) the probability that a random vector $w \in V_l$ falls into $V_l \cap V_k$ for some $k \neq l$,

4.) and the probability that there is a pair of secret subspaces V_l, V_k, $1 \leq l \neq k \leq L$, such that $\dim(V_l \oplus V_k) < n + k$.

For an estimation of the corresponding probabilities (for randomly chosen linear functions) see [KS09] and [KH11].

3.3 Impact of Splitting the Connection Function

In this section, we investigate any impact on the security caused by splitting the connection function. As the algebraic attack [KH11] (see Sec. 3.2 for a summary) is independent of the connection function, the resistance against this attack remains unchanged. However, as we elaborate below, the situation is different for the active MITM attack explained in [KS09]. This MITM attack has been called (x, y)-equality attack and was used to break, e.g., the CKK^2-protocol by Cichoń, Klonowski and Kutyłowski. We show that splitting the connection function implies an (for the attacker better) upper bound of about $2^{-n/4}$ for the success probability of this kind of attack against $(n, k, L)^{80}$-protocols. One consequence is that for the parameters suggested in section 4 (e.g., $n = 64$, $k = 32$, $L = 16$), a reasonable level of security can be reached by running the protocol a few times (e.g., four independently executed rounds would reduce the upper bound to $2^{-16 \cdot 4}$ if $n = 64$). The aim of an (x, y)-equality attacker Eve is to generate two messages $w \neq w' \in \mathrm{GF}(2)^{n+k}$ and to efficiently test by MITM-access to the protocol if w and $w \oplus w'$ belong to the same linear subspace V_l for some $l \in [L]$. As shown in [KS09], such an attack can be used to efficiently compute specifications of the subspaces $V_1, \ldots, V_L$. Eve works in three phases:

1. Send a message $y \in \mathrm{GF}(2)^N$ to Bob and receive $w' = F_l(f(y, b'))$.
2. Observe a challenge $a \in \mathrm{GF}(2)^N$ sent by Alice.
3. Compute a value $x = x(y, w', a) \in \mathrm{GF}(2)^N$, send it to Bob, receive the message $w = F_r(f(x, b))$ and send $w \oplus w'$ to Alice.

The success probability of the attack is given by the probability that Alice accepts $w \oplus w'$ if $l = r$.

The connection function of the $(n,k,L)^{80}$-protocol yields provable security against (x,y)-equality attacks. From now on we identify $\{0,1\}^4$ with the finite field $K = \mathrm{GF}(2^4)$ and denote by $+,\cdot$ the addition an multiplication in K. Let the function value $f(a,b)$ for all $a,b \in \{0,1\}^{n/2}$ be defined by $f(a,b) = \left(\left(a_1b_1, a_1b_1^3\right), \ldots, \left(a_{n/8}b_{n/8}, a_{n/8}b_{n/8}^3\right)\right)$, where $a_i, b_i \in K$, $i = 1,\ldots,n/8$, are obtained by partitioning a and b into blocks of 4 bits, respectively. Note that, according to the specification of the $(n,k,L)^{80}$-protocol (see subsection 2.2), the prover Bob will only reply to challenges a (and choose nonces b) which satisfy $a_i, b_i \neq 0$ for all $i = 1,\ldots,n/8$. Thus, Alice accepts a message w with $F_l^{-1}(w) = \left((u_1,v_1),\ldots,\left(u_{n/8},v_{n/8}\right)\right)$ in inner state $a \in (K^*)^{n/8}$ if for all $i = 1,\ldots,n/8$ it holds that $\left(a_i^{-1}u_i\right)^3 = a_i^{-1}v_i$, which is equivalent to $u_i^3 = a_i^2 v_i$.

Theorem 1 *The success probability of an (x,y)-equality attack against the $(n,k,L)^{80}$-protocol is at most $0.2^{n/8}$.*

Proof: See Appendix D.

4 Hardware Efficiency

Considered Metric. In order to assess the efficiency of our hardware implementation and to allow for comparing the results with other cryptographic protocols, generally accepted metrics are needed. Most authors consider *area*, *throughput* and *power consumption* the most important factors and, depending on the nature of their protocol, focus on one of them with respect to optimization (which is usually a trade-off). In the case of the suggested $(n,k,L)^{80}$-protocol, we will focus on area for the following reasons. Clearly, the throughput of our protocol is dominated by the speed of the keystream generator G (see, e.g., equation 1 in subsection 2.2). Given that G produces one bit every c clock cycles, computing the $n+k$ token bits $y_0 \ldots y_{n+k-1}$ takes $c \cdot n \cdot (n+k)$ clock cycles (after the initialization phase). Hence, for reasonably small values of c (e.g., $c = 4$ on average for the self-shrinking generator used in our implementation), the bottleneck w.r.t. to timing is not the speed at which the token is generated but rather the extremely limited transmission bandwidth of (passive) RFID tags. With respect to power consumption, the low clock rates of, e.g., 100 KHz in the context of RFID applications, lead to a situation where the static part of the power consumption becomes dominant. As this, in turn, can be decreased directly by minimizing the number of needed gates, it emphasizes our approach to focus on a low area footprint. In terms of hardware design, measuring the area size is a complicated task. First of all, one needs to distinguish between two main target platforms for our authentication protocol: *Field Programmable Gate Arrays* (FPGAs) and *Application-specific Integrated Circuits* (ASICs). As the names already suggest, FPGAs are integrated circuits designed to be configured by a customer or a designer after manufacturing, whereas ASICs are integrated circuits customized for a particular use, rather than intended for general-purpose use. Both worlds have rather different ideas of area, which will be explained, as needed, in subsections 4.1 and 4.2, respectively, along with the specific target devices and tools used.

Chosen Keystream Generator. The protocol can be instantiated with any secure and hardware efficient keystream generator. For our implementation we decided to use the self-shrinking generator [MS94] on top of a mere MLLFSR (a maximal length LFSR, i.e., an LFSR with a primitive feedback polynomial). While only few additional gates are needed to implement the logic of the self-shrinking generator as compared to a simple MLLFSR (see section 4.2), the security benefit is enormous. The best currently known attacks against self-shrinking generators are a time memory attack by Mihaljevic (1996) [Mih96] and an OBDD-attack by Krause (2001) [ZKL01]. However, we do not see how to use these attacks in order to realize a non-trivial attack against the $(n, k, L)^{80}$-protocol. In particular, the fact that no algebraic attacks are known makes the self-shrinking generator seem especially suited for our context.

General Remark. Before presenting our implementation results for FPGAs and ASICs in the following two sections, we would like to share our impression that despite the multitude of allegedly lightweight authentication protocols which have been suggested so far (see, e.g., [JW05], [BC08], [GRS08] or, more recently, [KPC$^+$11]), none of the respective works contains any of the cost metrics listed above. In contrast, newly introduced lightweight block ciphers like PRESENT [BKL$^+$07] or KATAN [DDK09] always come with an extensive assessment of their real-world hardware cost. This is why in subsections 4.1 and 4.2, we compare the numbers of $(n, k, L)^{80}$ rather with those of PRESENT, assuming its use as part of the following simple authentication scheme: Both parties share the encryption/decryption key of PRESENT as a common secret and in order to prove his identity, the prover needs to correctly encrypt a random nonce provided the verifier. However, we hope that our hardware results presented in this paper will encourage other designers of lightweight authentication protocols to also go trough the process of actually implementing their schemes in order to allow for easier efficiency comparison in the future.

4.1 The $(n, k, L)^{80}$-prover on FPGAs

In order to allow for an easy comparison on FPGAs, we implemented our authentication protocol for the *Spartan3 XC3S400* (Package FG456, Speed -5) from Xilinx [Xil13], using Verilog and their ISE Design Suite 14.1 for synthesis. Please refer to table 1 in subsection 1.2 for a concise overview of the corresponding implementation results. Clearly, while actually aimed at ASICs, the area footprint of the $(n, k, L)^{80}$-protocol is also very moderate on FPGAs, e.g., it amounts to 139 FFs (Flip Flops) and 177 4-input LUTs (Look-Up Tables) in the case of $n = 32$ and $k = 16$. This compares to 152 FFs and 253 LUTs given in [Pos09] for the encryption unit of PRESENT-80 on the same platform and an *espresso*-optimized [UoC94] S-box. Without this latter optimization, the respective numbers are 154 FFs and 350 LUTs.

Overall, these numbers suggest, that our preliminary implementation of the $(n, k, L)^{80}$-protocol is already a viable alternative to the optimized code of PRESENT when it comes to authentication schemes. The subsequent section will even reinforce this impression.

4.2 The $(n, k, L)^{80}$-prover on ASICs

ASICs are a typical component in the context of RFID applications. They are (ex ante) tailored to a very specific need and subsequently produced in large quantities, allowing for low unit cost and making them perfectly suitable for pervasive devices like RFID tags. In the field of ASICs, area is usually measured in μm^2. However, as area requirements in μm^2 strongly depend on the used standard cell library (and, thus, the fabrication technology), it is common to use a metric called Gate Equivalents (GEs) instead. In short, one GE is equivalent to the area of a two-input drive-strength-one NAND gate. This at least allows for a rough comparison of area requirements derived using different technologies. As in the case of FPGAs, we again chose a technology closely related to the one which was used to derive the corresponding results for PRESENT-80 in [Pos09] to allow for a fair comparison. Synthesis and analysis was performed using *Cadence Encounter RTL Compiler RC11.24* [Cad13] and the employed technology library was *UMCL18G212T3*.

For the given technology and the parameter choice $(n, k, L) = (32, 16, 32)$, an ASIC implementation of the $(n, k, L)^{80}$-prover requires 1281 GEs. This is well below 2000 GEs, commonly referred to as the maximum area available on an RFID device for cryptographic purposes. In comparison, according to [Pos09], implementing PRESENT-80 in a fast round-based manner takes 1570 GEs, which is about the same size as the 1565 GEs needed to realize the $(n, k, L)^{80}$-prover for large parameters, i.e., $(n, k, L) = (64, 32, 16)$.

5 Conclusion

We introduced the $(n, k, L)^{80}$ authentication protocols, which are a modification of the already investigated $(n, k, L)^{++}$-protocol made in order to improve hardware efficiency. Our implementations confirm the suitability of our protocols for use cases which demand for low hardware size, e.g., RFID systems, making them interesting for practice. Moreover, the fact that the security of these protocols relies on a different paradigm than the alternative approaches based on block ciphers or the LPN problem, i.e., the random selection of secret functions, makes this kind of protocols likewise interesting for the cryptography community. One major modification is that the internal linear functions are generated by a bitstream generator G in order to save memory. Our analysis shows that, while using a single publicly known LFSR renders the protocol insecure, deploying a secure pseudorandom bit generator is sufficient. However, it remains an open question whether other, intermediate approaches, e.g., using an NLFSR or possibly keeping the LFSR-specifications secret, might be viable alternatives. In general, given that the underlying problem is relatively new, its hardness and possible connections to other problems need to be investigated further. Moreover, despite the popularity of lightweight authentication protocols, it turns out that only few actual implementations exist, which, in addition, commonly shift the problem (and cost) of generating random bits on the prover's side to a higher level of the tag's hardware (as we do in this work, too). This aspect represents an important next step towards a better understanding and comparison of existing design approaches.

References

[BC08] J. Bringer and H. Chabanne. Trusted-HB: A low cost version of HB$^+$ secure against a man-in-the-middle attack. *IEEE Trans. Inform. Theor.*, 54:4339–4342, 2008.

[BKL+07] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. H. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *Proceedings of Cryptographic Hardware and Embedded Systems (CHES) 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[BKM+09] E.-O. Blass, A. Kurmus, R. Molva, G. Noubir, and A. Shikfa. The F_f-Family of Protocols for RFID-Privacy and Authentication. In *5th Workshop on RFID Security, RFIDSec'09*, 2009.

[Cad13] Cadence. Encounter RTL Compiler, 2013. `http://www.cadence.com/products/ld/rtl_compiler/`.

[CKo08] J. Cichoń, M. Klonowski, and M. Kutyłowski. Privacy Protection for RFID with Hidden Subset Identifiers. In *Proceedings of Pervasive 2008*, volume 5013 of *LNCS*, pages 298–314. Springer, 2008.

[DDK09] C. De Cannière, O. Dunkelman, and M. Knežević. KATAN and KTANTAN – A Family of Small and Efficient Hardware-Oriented Block Ciphers. In *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES) 2009*, volume 5747 of *LNCS*, pages 272–288. Springer, 2009.

[FS09] D. Frumkin and A. Shamir. Untrusted-HB: Security Vulnerabilities of Trusted-HB. Cryptology ePrint Archive, Report 2009/044, 2009. `http://eprint.iacr.org`.

[GRS05] H. Gilbert, M. J. B. Robshaw, and H. Sibert. Active Attack against HB$^+$: A provable secure lightweight authentication protocol. *Electronic Letters*, 41:1169–1170, 2005.

[GRS08] H. Gilbert, M. J. B. Robshaw, and Y. Seurin. HB$^\#$: Increasing the Security and Efficiency of HB$^+$. In *Proceedings of Eurocrypt 2008*, volume 4965 of *LNCS*, pages 361–378, 2008.

[HKL+12] Stefan Heyse, Eike Kiltz, Vadim Lyubashevsky, Christof Paar, and Krzysztof Pietrzak. Lapin: An Efficient Authentication Protocol Based on Ring-LPN. In Anne Canteaut, editor, *Fast Software Encryption*, volume 7549 of *Lecture Notes in Computer Science*, pages 346–365. Springer Berlin Heidelberg, 2012.

[JW05] A. Juels and S. A. Weis. Authenticating Pervasive Devices with Human Protocols. In *Proceedings of Crypto 2005*, volume 3621 of *LNCS*, pages 293–308. Springer, 2005.

[KH11] M. Krause and M. Hamann. The Cryptographic Power of Random Selection. In *Proceedings of SAC 2011*, volume 7118 of *LNCS*, pages 134–150. Springer, 2011.

[KMNP11] S. Knellwolf, W. Meier, and M. Naya-Plasencia. Conditional Differential Cryptanalysis of Trivium and KATAN. In *Proceedings of SAC 2011*, volume 7118 of *LNCS*, pages 200–212. Springer, 2011.

[KPC+11] E. Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, and Daniele Venturi. Efficient authentication from hard learning problems. In *Proceedings of Eurocrypt 2011*, volume 6632 of *LNCS*, pages 7–26. Springer, 2011.

[KS09] M. Krause and D. Stegemann. More on the Security of Linear RFID Authentication Protocols. In *Proceedings of SAC 2009*, volume 5867 of *LNCS*, pages 182–196. Springer, 2009.

[Mih96] M. Mihaljević. A faster cryptanalysis of the self-shrinking generator. In *Information Security and Privacy*, volume 1172 of *LNCS*, pages 182–189. Springer, 1996.

[MS94] Willi Meier and Othmar Staffelbach. The Self-Shrinking Generator. In *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 of *Lecture Notes in Computer Science*, pages 205–214. Springer, 1994.

[OOV08] K. Ouafi, R. Overbeck, and S. Vaudenay. On the Security of HB$^{\#}$ against a Man-in-the-middle Attack. In *Proceedings of Asiacrypt 2008*, volume 5350 of *LNCS*, pages 108–124. Springer, 2008.

[Pos09] Axel York Poschmann. Lightweight Cryptography: Cryptographic Engineering for a Pervasive World, 2009.

[Å11] M. Ågren. Some Instant- and Practical-Time Related-Key Attacks on KTANTAN32/48/64. In *Proceedings of SAC 2011*, volume 7118 of *LNCS*, pages 213–229. Springer, 2011.

[UoC94] Berkeley University of California. Espresso, 1994. `http://embedded.eecs.berkeley.edu/pubs/downloads/espresso/index.htm`.

[Xil13] Xilinx. Programmable Devices and Design Resources, 2013. `http://www.xilinx.com/`.

[ZKL01] E. Zenner, Matthias Krause, and Stefan Lucks. Improved Cryptanalysis of the Self-Shrinking Generator. In *Information Security and Privacy*, volume 2119 of *LNCS*, pages 21–35. Springer, 2001.

A Implementation Results

Parameters				FPGA		ASIC		Comm. (Tag)
n	k	L	Rnds.	Slice-FFs	LUTs	GEs (μm^2)	Clk.	Bits IN/OUT
64	32	16	2	175	205	1,565 (15,147)	31,210	64/192
32	16	32	4	139	177	1,281 (12,402)	11,540	64/192

Table 1: An overview of the results of our hardware implementation for FPGAs (Xilinx Spartan3 XC3S400) and ASICs. *Clk.* denotes the total number of clock cycles needed on the prover's side to perform a full authentication consisting of multiple rounds. Due to the nature of the self-shrinking generator used in our implementation, the timing values in the respective column may vary slightly for different keys. (see section 2 for an in-depth explanation of the given parameters and section 4 for further details relating hardware costs)

B The $(n, k, L)^{++}$-protocol

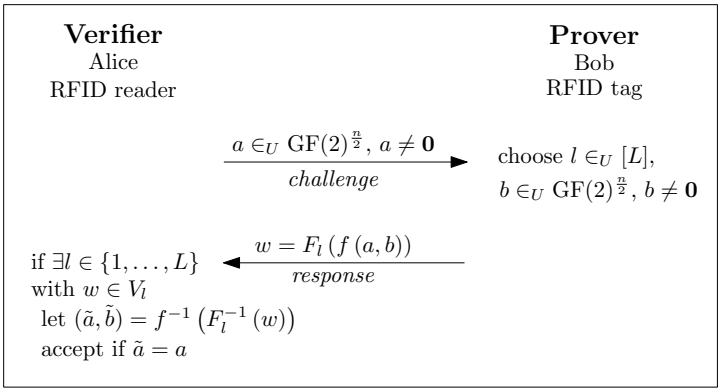


Figure 1: An instance of the $(n, k, L)^{++}$-protocol (cf. [KS09]).

C Algebraic Attack for Weak Generators.

In the following, we present an efficient algebraic attack if the generator G produces a bitstream with a short, known linear complexity. For simplicity, we consider the case that

G is realized by an MLLFSR (a maximal length LFSR, i.e., an LFSR with a primitive feedback polynomial). Observe however that the same attack works against any G which produces a bitstream with low, known linear span.

We first recall the Passive Algebraic Attack against $(n,k,L)^{++}$-type protocols presented in [KH11]. Building on this, we then show that generating the function matrices F_l by a single LFSR is not a good idea as this allows for a much more efficient attack. Due to the pseudorandomness assumption regarding the generator G formulated above, we assume that it is not possible to algebraically attack the $(n,k,L)^{80}$-protocol in a significantly more efficient way than the original $(n,k,L)^{++}$-protocol.

Let $F_1,\ldots,F_L : GF(2)^n \longrightarrow GF(2)^{n+k}$ denote the secret key consisting of L injective $GF(2)$-linear mappings, where k,n,L are appropriately chosen. During a passive key recovery attack the attacker tries to compute specifications of these function on the basis of pairs (x,y), where x is randomly and uniformly chosen from $GF(2)^n$ and it holds that $y = F_l(x)$ for some index l, which is randomly and uniformly chosen from $\{1,\ldots,L\}$.

The passive attack described in [KH11] is based on choosing appropriate parameters λ,μ such that $\lambda \cdot \mu = n+k$, considering the secret functions F_l as vectors of μ component functions mapping from $GF(2)^n$ into $GF(2)^\lambda$, identifying $GF(2)^\lambda$ with the finite field $K = GF(2^\lambda)$, and computing the component functions by means of the following algebraic attack approach:

Suppose we are given secret functions $f_1,\ldots,f_L : K^n \longrightarrow K$ and we want to compute specifications of these functions on the basis of known plaintext pairs (x,y), where x is randomly and uniformly chosen from $\{0,1\}^n \subseteq K^n$ and it holds that $y = f_l(x)$ for some secret index l, which is randomly and uniformly chosen from $\{1,\ldots,L\}$.

We were done if we could compute the values $x_{i,l} = f_l(e_i)$ for $i = 1,\ldots,n$ and $l = 1,\ldots,L$, where $e_i \in K^n$ denotes the standard vector having one at position i and zero at all other positions.

Note that each known plaintext pair (x,y) yields a degree-L equation in the $x_{i,l}$-variables of the form

$$\prod_{l=1}^{L}\left(\bigoplus_{i\in I} x_{i,l} \oplus y\right) = 0,$$

where $x = \bigoplus_{i\in I} e_i$.

In [KH11] it is shown that systems built of degree-L equations of this kind can be solved by a nontrivial application of the technique of linearization, which implies to solve a system of linear equations over $O(n^L)$ variables.

We analyze now the case that the $((n+k)\cdot n)\cdot L$ key bits characterizing the secret linear functions $F_1,\ldots,F_L$ are generated by one MLLFSR of length $m+M$, where $L = 2^M$. Remember that the secret symmetric key $\kappa = (\kappa_1,\ldots,\kappa_m)$ and the M random bits $l_1,\ldots,l_M$ forming the binary representations of the indexes $l \in \{1,\ldots,L = 2^M\}$ serve as the initial state of the LFSR.

We show in the following that this construction opens the door to an algebraic attack allowing to compute the secret key bits much more efficiently as compared to the general

case described in [KH11].

For demonstrating this we consider the algebraic attack of [KH11] against general linear protocols described above and suppose that λ is chosen by the attacker such that $\lambda = M + 1$. Our construction implies that each bit of the function matrices of $F_1, \ldots, F_L$, and consequently each bit of the secret K-elements $x_{i,l}$, is the output of a publicly known $GF(2)$-linear mapping in the k-bits and the random l-bits.

Hence, the secret K-elements $x_{i,l}$ can be written as

$$x_{i,l} = \bigoplus_{s=1}^{m} c_{i,s} k_s \oplus \bigoplus_{t=1}^{M} C_{i,t} l_t,$$

where $\tilde{l} = (l_1, \ldots, l_M)$ and the vectors $c_{i,s}, C_{i,t} \in GF(2)^\lambda$ are publicly known. Thus, each known plaintext pair (x, y), $x = \bigoplus_{i \in I} e_i$, translates into the statement that

$$\bigoplus_{s=1}^{m} \left(\bigoplus_{i \in I} c_{i,s} \right) k_s \ \in \ W(y),$$

where the set $W(y) \subseteq GF(2)^\lambda$ is defined by $W(y) = \{y \oplus C_{I,1}, \ldots, C_{I,L}\}$ and for each $\tilde{l} = (l_1, \ldots, l_M)$ representing an element of $\{1, \ldots, L\}$ it holds that

$$C_{I,l} = \bigoplus_{t=1}^{M} \left(\bigoplus_{i \in I} C_{i,t} \right) l_t.$$

Now we can compute a nonzero Boolean function $g : \{0,1\}^\lambda \longrightarrow \{0,1\}$ which annihilates $W(y)$. This is possible as $W(y)$ is a proper subset of $\{0,1\}^\lambda$ due to $|W(y)| \leq 2^M = 2^{\lambda-1}$.

More concretely, we compute a square free polynomial $p = p(z_1, \ldots, z_\lambda)$ which yields g. This can be done by solving a system of at most L $GF(2)$-linear equations in at most 2^λ variables corresponding to the square free monomials over $z_1, \ldots, z_\lambda$. As M and λ are small numbers in practice, this is feasible. Note that the degree of p is at most λ.

Consequently, the known plaintext pair (x, y) yields the following nonlinear equation in the key bits:

$$p\left(\bigoplus_{s=1}^{m} \left(\bigoplus_{i \in I} c_{i,s} \right) k_s \right) = 0.$$

The degree of this equation is at most $\lambda = \log_2(L) + 1$, which is much smaller than L, the degree of the algebraic attack for the general case.

D Proof of Theorem 1

Proof: For given $y, a \in (K^*)^{n/8}$, Eve has to choose an element $x \in (K^*)^{n/8}$ such that

$$w + w' = F_l\left((u_1, v_1), \ldots, \left(u_{n/8}, v_{n/8}\right)\right)$$

will be accepted by Alice in inner state a, where $w = F_l\left(f\left(x, b\right)\right)$ and $w' = F_l\left(f\left(y, b'\right)\right)$ for some $l \in [L]$, and $b, b' \in \left(K^*\right)^{n/8}$. Note that Eve has no information about b, b', and that $u_i = x_i b_i + y_i b'_i$ and $v_i = x_i b_i^3 + y_i b_i'^3$ for $i = 1, \ldots, n/8$.

Consequently, Eve's choice for the value x has to satisfy

$$\left(x_i b_i + y_i b'_i\right)^3 = a_i^2 \left(x_i b_i^3 + y_i b_i'^3\right)$$

for all $i = 1, \ldots, n/8$.

This is equivalent to

$$\left(x_i + y_i c_i\right)^3 = a_i^2 \left(x_i + y_i c_i^3\right),$$

where $c_i = b'_i \left(b_i^{-1}\right)$, which, in turn, is equivalent to $P_i\left(x_i, c_i\right) = 0$, where the polynomial $P_i\left(x_i, d_i\right)$ is for all $d_i \in K^*$ and $i = 1, \ldots, n/8$ defined as

$$P_i\left(x_i, d_i\right) = x_i^3 + \left(y_i d_i\right) x_i^2 + \left(y_i^2 d_i^2 + a_i^2\right) x_i + d_i^3 \left(y_i^3 + y_i a_i^2\right).$$

Note that there are $|K^*| = 15$ different polynomials of type $P\left(x_i, d_i\right)$ with respect to the variable x_i (look at the coefficient $y_i d_i$ of x_i^2).

For all $x_i \in K^*$ let $P_i\left(x_i\right) = \left\{d_i \mid P\left(x_i, d_i\right) = 0\right\}$. $P_i\left(x_i, d_i\right)$ is a polynomial of degree 3 also in the unknown d_i, which implies $\left|P_i\left(x_i\right)\right| \leq 3$ for all $x_i \in K^*$.

Eve has to choose an $x \in \left(K^*\right)^{n/8}$ that satisfies $c_i \in P_i\left(x_i\right)$ for all $i = 1, \ldots, n/8$. Since she does not have any information about $c_1, \ldots, c_{n/8}$, her success probability is bounded from above by

$$\prod_{i=1}^{n/8} \frac{3}{15} = 0.2^{n/8}.$$

This concludes the proof. □

Chameleon-Hashing für Interaktive Beweise der Verfügbarkeit dynamischer Daten in der Cloud*

Stefan Rass, Peter Schartner

Forschungsgruppe Systemsicherheit, Institut für Angewandte Informatik
Alpen-Adria Universität Klagenfurt
Universitätsstraße 65-67
9020 Klagenfurt
stefan.rass@aau.at
peter.schartner@aau.at

Abstract: Proofs of Retrievability (PoR) sind interaktive Verfahren, welche die konsistente Existenz von Daten, im Kontext von Cloud-Storage-Diensten, überprüfen lassen. Der triviale Ansatz durch Download und Überprüfung des gesamten Datenvolumens ist bei großen Datenmengen nicht praktikabel, und PoR-Verfahren verfolgen das Ziel, diese Überprüfung wesentlich effizienter und mit geringem Aufwand für den Kunden zu ermöglichen. Dynamische PoR-Verfahren bieten überdies die Möglichkeit, die Daten am Server nachträglich zu verändern. In der vorliegenden Arbeit beschreiben wir eine einfache und effiziente Konstruktion eines dynamischen proof of retrievability auf Basis von Chameleon-Hashfunktionen.

1 Einleitung

Eine häufig genutzte Form von Dienstleistungen im Rahmen von Cloud Computing ist das Angebot von Speicherplatz für Zwecke der Archivierung bzw. des Austausches großer Datenmengen. Hierbei steht neben Vertraulichkeit insbesondere Verfügbarkeit und Integrität im Zentrum der Interessen der Kunden. Im einfachsten Fall können beide Eigenschaften durch den Download und die lokale Überprüfung des gesamten Datenbestandes beim Kunden sichergestellt werden. Da dieses Vorgehen für große Datenmengen ineffizient und nicht praktikabel ist, wurden von [JK07, LEB+03] interaktive Protokolle zum Nachweis von Integrität und Verfügbarkeit großer Datenmengen vorgeschlagen, sogenannte *proofs of retrievability* (PoR). Hierbei handelt es sich um Challenge-Response-Protokolle, welche unter Zuhilfenahme einer geringen Menge von Verifikationsdaten (welche der Kunde speichert), effiziente Überprüfungen von Verfügbarkeit und Integrität großer Datenbestände ermöglichen.

*Die vorliegende Arbeit ist eine modifizierte und erweiterte Fassung des Artikels "Dynamic Proofs of Retrievability from Chameleon-Hashes", erschienen in: Proceedings of the 10th International Conference on Security and Cryptography (SECRYPT), IEEE, 2013, pp. 296-304. [Ras13]

Die im Folgenden verwendete Terminologie ist jener von interaktiven Beweisen angepasst: der *Verifizierer* (*Client*) ist der Kunde, welcher seine Datenbestände bei einem externen (Cloud) Provider (auch *Beweiser, Prover* oder *Server* genannt), speichert. Ein PoR-Protokoll ist ein Beweis von Wissen (proof of knowledge), in dem Sinne, dass ein Wissensextraktor (knowledge extractor; Algorithmus bzw. Protokoll) existiert, welcher aus den Antworten des Servers den gesamten Datenbestand rekonstruieren kann. Obgleich diese Funktionalität wesentlich der Sicherstellung theoretischer Eigenschaften von PoR-Verfahren dient, kann die Wissensextraktion intuitiv als gewöhnlicher "Download" der Daten betrachtet werden.

Im einfachsten Fall kann eine Challenge-Response-Überprüfung der Verfügbarkeit einer Datei F wie folgt ablaufen: der Client sendet einen (zufälligen) Schlüssel k an den Server, welcher mit einem Message-Authentication Code $MAC_k(F)$ der Datei F antwortet. Diesen MAC kann der Client mit einem bei ihm lokal gespeicherten Wert vergleichen (eine endliche Liste gültiger MACs wird *vor* Ausführung des Protokolls berechnet und beim Client gespeichert), um die Integrität und Verfügbarkeit der Daten zu überprüfen. Dieses sehr einfache Verfahren besitzt mehrere Nachteile, welche durch verbesserte Verfahren behoben werden:

N1: Das Protokoll lässt nur eine begrenzte Anzahl an Überprüfungen zu, da die Verifikationsdatenmenge beim Client auf einen kleinen Bruchteil der Originaldatenmenge begrenzt sein muss (andernfalls wäre es effizienter, die gesamten Daten beim Client zu belassen).

N2: Der Server muss für die Antwort den gesamten Datenbestand verarbeiten, was im Falle großer Datenmengen zu langen Antwortzeiten führen kann.

N3: Die Daten dürfen sich über ihre gesamte Lebensdauer nicht verändern, da andernfalls die beim Client gespeicherten Antworten ungültig werden würden.

Insbesondere Eigenschaft N3 führt für diese Methode zu der Bezeichnung *statisches* PoR-Verfahren. Diese Einschränkung ist vielen in jüngerer Zeit verfügbaren PoR-Verfahren gemeinsam. Im Gegensatz dazu erlauben *dynamische* PoR-Verfahren auch nachträgliche Veränderungen der Daten. Wir beschreiben nachfolgend ein solches dynamisches PoR-Verfahren, welches explizit alle drei genannten Mängel behebt.

Eine weitere Klassifizierung von PoR-Verfahren kann auf Basis von Eigenschaft N1 getroffen werden: hierbei wird zwischen *schlüsselabhängigen* und *schlüssellosen* Verfahren unterschieden. Letztgenannte erlauben nur eine feste (beschränkte) Anzahl von Wiederholungen (d.h. Eigenschaft N1 gilt), während schlüsselabhängige Verfahren i.d.R. eine unbegrenzte Anzahl an Challenge-Response-Verifikationen gestatten (Eigenschaft N1 gilt nicht). Hierbei bestehen enge theoretische Beziehungen zu dem verwandten Konzept des beweisbaren Datenbesitzes (*provable data possession* PDP [ABC$^+$07]). Auch diese Verfahren existieren in statischen und dynamischen Ausprägungen ([ADPMT08, EKPT09]), werden im Allgemeinen jedoch als schwächer als PoR angesehen, da der Wissensextraktor bei einem PDP Verfahren nicht gefordert wird (obgleich dieser implizit im Rahmen der Sicherheitsbeweise dennoch oft zu finden ist). Die genauen Verbindungen zwischen PoR und

PDP sind aktuell noch Gegenstand laufender und intensiver Untersuchungen, und werden im Weiteren nicht näher diskutiert.

Verwandte Arbeiten: Das erste als solches bezeichnete PoR-Verfahren wurde in [JK07] vorgeschlagen, wobei die Grundideen bereits in [LEB+03] skizziert wurden. Insbesondere Eigenschaft N1 motivierte Vorschläge schlüsselabhängiger Verfahren [SW08, XC12], welche eine unbegrenzte Anzahl an Verifikationszyklen gestatten. Die Arbeit von [PSU12] zeigt eine theoretische Äquivalenz zwischen PoR-Verfahren und fehlerkorrigierenden Codes auf, welche als Bausteine fast aller PoR-Verfahren zum Einsatz kommen, insbesondere da der Angreifer als Kanal mit Rauschen modelliert werden kann (etwa in [BJO09]). Das nachfolgend angegebene Verfahren ist berechenmäßig sicher; informationstheoretisch sichere Verfahren wurden u.a. in [DVW09] vorgeschlagen. Dynamische Varianten von PoR-Verfahren sind in [ZX11, CKW12, CC12] zu finden, wobei hier insbesondere weitere Sicherheitsanforderungen eingeführt werden, wie etwa Robustheit (Wiederherstellbarkeit auch im Lichte umfangreicher Modifikationen der Daten) oder Fairness (Sicherheit des Servers gegen unrechtmäßigen Beschuldigungen des Clients, dass die Daten manipuliert wurden). An dieser Stelle sei insbesondere auf die Arbeit von [WWR+11] hingewiesen, welcher Aktualisierungen der Daten mit Hilfe von Merkle-Hashbäumen realisiert. Wir werden ähnliche Techniken anwenden, sind jedoch flexibler als [WWR+11] im Hinblick auf die verwendbaren Krypto-Verfahren. Da hier beschriebene Technik ist in weiten Teilen generisch und kann mit verschiedenen Verfahren instantiiert werden. Neben PDP existieren auch die verwandten Konzepte des *proof of storage* [AKK09] und des *proof of ownership* [HHPSP11], auf welche wir nicht näher eingehen.

Beiträge und Konstruktion: Aktuelle PoR-Verfahren können im Allgemeinen auf Basis von Stichprobenüberprüfungen oder unter Ausnützung von Homomorphie-Eigenschaften [LC11] konstruiert werden. Bei einer stichprobenbasierten Überprüfung bettet der Client Prüfdaten, sog. *Marker* (engl. *sentinels*), in die Datei ein, deren Wert später im Rahmen einer PoR-Überprüfung abgefragt werden kann. Diese PoR-Instanzierungen sind im Allgemeinen schlüssellos und unterscheiden sich im Wesentlichen in der Art und Weise, wie diese Marker in die Datei eingebettet werden, um deren Position und Inhalt vor einem nicht vertrauenswürdigen Server zu verbergen. PoR-Instanzierungen auf Basis von Homomorphie-Eigenschaften von digitalen Signaturen oder MACs sind schlüsselabhängig und erfordern häufig die Verarbeitung der gesamten Datenmenge zur korrekten Beantwortung der Anfrage. Der Vorteil einer unbeschränkten Anzahl von Verifikationszyklen wird somit durch erhöhten Rechenaufwand erkauft.

Das in diesem Beitrag vorgestellte Verfahren fällt nicht direkt in eine dieser beiden Kategorien. Die Konstruktion basiert auf Stichproben-Überprüfungen, bettet jedoch keine Verifikationsdaten in die Datei ein, und gestattet – anders als andere Vertreter dieser Kategorie – auch ein nachträgliches Erweitern der lokalen Verifikationsdaten beim Client.

Die nachfolgend beschriebene Konstruktion verfolgt im Wesentlichen die bereits in Abschnitt 1 skizzierte Idee. Der Client sendet dem Server eine Anfrage (challenge) c, welche eine bestimmte Modifikation der Daten F am Server beschreibt. Der Server antwor-

tet mit dem Hashwert der durch c modifizierten Daten F', welche der Client mit seinen lokal gespeicherten Verifikationsdaten vergleicht. Zur Effizienzsteigerung berechnet der Server den Hashwert der gesamten Daten mit Hilfe eines Merkle-Hashbaumes (Eigenschaft N2 fällt somit als Nachteil weg). Modifikationen an den Daten F sind für den Client nur dann möglich, wenn diese den Hashwert des betreffenden Datensatz nicht verändern. Dies kann durch Einsatz von Chameleon-Hashfunktionen realisiert werden (Einschränkung/Eigenschaft N3 fällt weg), welche insbesondere dann auch erneute Abfragen des Datensatzes unter anderen Modifikationen gestatten. Das Verfahren ermöglicht damit eine unbegrenzte Anzahl an Modifikationen der Datei F, und aus diesem Grund auch eine quasi unbegrenzte Anzahl an Verifikationszyklen (Eigenschaft/Nachteil N1 ist somit hinfällig).

Abschnitt 2 stellt die im Weiteren benötigten Konzepte zur Verfügung, bevor wir uns der detaillierten Konstruktion in Abschnitt 3 widmen.

2 Definitionen

Wir nennen eine Funktion $\nu(t)$ *vernachlässigbar*, wenn $\nu(t) < 1/|Q(t)|$ für jedes Polynom Q und alle hinreichend großen $t \in \mathbb{N}$. Eine Wahrscheinlichkeit p nennen wir *überwältigend*, wenn $1 - p$ vernachlässigbar ist. Die Notation $x\|y$ stellt eine Codierung von x und y als String dar, aus welcher x und y eindeutig rekonstruierbar sind. Wir nehmen o.B.d.A. an, dass eine Datei $F = x_1\|x_2\|\ldots\|x_n$ als Folge von Datensätzen, *Blöcken*, x_i, $i = 1, 2, \ldots, n$, angesehen werden kann. An dieser Stelle sei darauf hingewiesen, dass das nachfolgend beschriebene Verfahren sich in kanonischer Weise auf baumstrukturierte Daten (etwa XML-Dateien) übertragen lässt, und – anders als andere PoR-Verfahren – keine festgelegte Datenstruktur voraussetzt. Insbesondere kann die Partitionierung der Daten F in Blöcke beliebig und sinngemäß, etwa durch ein XML-Schema, festgelegt werden.

2.1 Struktur eines PoR-Verfahrens

Wir verwenden die für statische PoR typische Definition [JK07], welche wir entsprechend um Funktionalität für Datenaktualisierungen erweitern. Ein proof of retrievability besteht (im Allgemeinen) aus folgenden Komponenten:

`Setup`(t)**:** Ein probabilistischer Algorithmus, welcher mit einem Sicherheitsparameter $t \in \mathbb{N}$ sämtliche beteiligten kryptographischen Systeme initialisiert und die erforderlichen Schlüssel und Systemparameter zurückgibt. Diese werden allen nachfolgenden Algorithmen zur Verfügung gestellt, und in der Auflistung der Parameter nicht mehr explizit genannt.

`Encode`(F)**:** Algorithmus, welcher die Daten F entgegennimmt, und in einer (fehlerkorrigierenden) Art und Weise codiert, sodass spätere Challenge-Response-Verifikationszyklen ermöglicht werden. Im Allgemeinen können sowohl der Client als auch der

Server eine Fehlerkorrektur durchführen (um etwa Übertragungsfehlern entgegenzuwirken), wobei wir nachfolgend davon ausgehen, dass die Fehlerkorrektur implizit geschieht und wir hierauf nicht weiter eingehen werden. Das Ergebnis von `Encode` ist die codierte Datei F^* und die Information β, aus welcher sich die späteren Anfragen und Verifikationsdaten ableiten lassen.

`Challenge`(β)**:** Der Algorithmus $\mathcal{V}_{\text{chal}}$ erzeugt aus den Daten β eine Anfrage (challenge) c.

`Response`(c, β)**:** Erzeugt aus der Anfrage c die zugehörige Antwort r.

`Verify`(c, r, β)**:** Der Algorithmus $\mathcal{V}_{\text{verify}}$ überprüft ein gegebenes Challenge-Response-Paar (c, r) auf Korrektheit, und liefert entweder 1 (korrekt) oder 0 (inkorrekt).

`Update`: Das interaktive Protokoll $\mathcal{V}_{\text{upd}}$ zwischen dem Client und dem Server ersetzt einen durch den Index i adressierten Datensatz x_i mit einem neuen Wert $\widetilde{x}_i$. Gleichzeitig werden die Daten β beim Client aktualisiert.

`Extract`(β)**:** Algorithmus, welcher eine Sequenz von Anfragen (challenges) erzeugt, aus welchen die Daten F' rekonstruiert werden (entspricht dem Download der gesamten Datenmenge).

Angreifer und Sicherheitsmodell: Wir verwenden das in der Originalarbeit [JK07] vorgeschlagene Sicherheitsmodell in einer geeigneten Erweiterung. Der *Angreifer* $\mathcal{A}$ ist ein Paar von probabilistischen Algorithmen $\mathcal{A}_{\mathsf{setup}}, \mathcal{A}_{\mathsf{resp}}$. Algorithmus $\mathcal{A}_{\mathsf{setup}}$ interagiert mit dem Client (Verifizierer) $\mathcal{V}$ um das PoR-Protokoll zu initialisieren. Insbesondere erzeugt der Angreifer (Server) eine Datei F^*, welche er lokal speichert und dem Verifizierer (Client) die für die Verifikation benötigten Daten β und PoR-Systemparameter α zur Verfügung stellt. Während eines Challenge-Response-Zyklusses berechnet der Angreifer mit dem Algorithmus $\mathcal{A}_{\mathsf{resp}}$ die erforderlichen Antworten. Das Verfahren wird durch eine Ausführung von `Extract` beendet, wobei der Client die Datei F' rekonstruiert. Wir betrachten einen Angriff als erfolgreich, wenn der Client $F' \neq F^*$ erhält.

Es bezeichne $t \in \mathbb{N}$ den Sicherheitsparameter und α die Systemparameter. Die Schreibweise $\mathcal{A}^{\mathcal{B}}$ notiert Orakel-Zugriff (bzw. Interaktion) des Algorithmus $\mathcal{A}$ auf (bzw. mit) Algorithmus $\mathcal{B}$. Mit $x \in_R X$ bezeichnen wir das gleichverteilt zufällige Ziehen eines Elements $x \in X$. Die vorhin beschriebenen Abläufe sind als folgende Experimente formalisierbar:

Experiment $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{setup}}(t)$	Experiment $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha)$
$\kappa \leftarrow \mathtt{KeyGen}(t)$	$\mathsf{action} \leftarrow_R \{\mathsf{chal}, \mathsf{upd}\}$
$(F^*, \alpha) \leftarrow \mathcal{A}^{\mathcal{V}}_{\mathsf{setup}}$	$c \leftarrow \mathcal{V}_{\mathsf{action}}(\alpha)$
übergebe α an $\mathcal{V}$	$r \leftarrow \mathcal{A}_{\mathsf{resp}}(F^*, \alpha, c)$
	return $\mathcal{V}_{\mathsf{verify}}(r, \alpha)$

Wir betrachten ein PoR-Protokoll als *sicher*, wenn ein Angreifer, welcher in Experiment $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha)$ mit überwältigender Wahrscheinlichkeit $\geq 1 - \zeta$ erfolgreich ist, den Verifizierer *nicht* dazu bringen kann, etwas anderes als F^* herunterzuladen (durch `Extract`).

Anders ausgedrückt soll Erfolg im Experiment $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha)$ (also das korrekte Antworten auf Challenges) auch nachfolgend die Rekonstruktion der korrekten Daten F^* implizieren. Die Erfolgswahrscheinlichkeit in $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha)$ wird bezeichnet mit

$$\mathbf{Succ}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha) := \Pr\left[\mathbf{Exp}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha) = 1\right].$$

Das Sicherheitsmodell ist nun wie folgt festgelegt: Der Angreifer $\mathcal{A}$ besitze die Datei F^*, welche im Rahmen von $\mathbf{Exp}^{\mathcal{A}}_{\mathsf{setup}}(t)$ erzeugt wurde. Der Client $\mathcal{V}$ interagiert mit $\mathcal{A}_{\mathsf{resp}}$ und führt Challenge-Response-Zyklen durch, mit einem abschließenden Aufruf von `Extract`. Der Angriff gilt als erfolgreich, wenn der Client $F \neq F^*$ als Ergebnis erhält. Die Wahrscheinlichkeit, dass dies *nicht* passiert, ist

$$\mathbf{Succ}^{\mathcal{A}}_{\mathsf{extract}}(F^*, \alpha) := \Pr\left[F = F^* | F \leftarrow \mathtt{extract}^{\mathcal{A}_{\mathsf{resp}}}(\alpha)\right].$$

Definition 2.1 *Wir nennen ein PoR-Verfahren (ρ, λ)-sicher, wenn eine im Sicherheitsparameter t vernachlässigbare Funktion $\zeta(t)$ existiert, sodass*

$$\Pr\left[\begin{array}{c|l} \mathbf{Succ}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha) \geq \lambda, & (F^*, \alpha) \leftarrow \mathbf{Exp}^{\mathcal{A}}_{\mathsf{setup}}(t), \\ \mathbf{Succ}^{\mathcal{A}}_{\mathsf{extract}}(F^*, \alpha) < 1 - \zeta & F \leftarrow \mathtt{extract}^{\mathcal{A}_{\mathsf{resp}}}(\alpha) \end{array}\right] \leq \rho.$$

Man beachte, dass die Parameter ρ und λ gegenläufig sind. Wir sind an Verfahren mit *großen* Werten für λ und *kleinen* Werten für ρ interessiert. In solchen Fällen gilt: mit hoher Wahrscheinlichkeit ($> 1 - \rho$) können die Daten F korrekt heruntergeladen werden, oder andernfalls wird die Manipulation durch den Server im Zuge eines Challenge-Response-Verifikationszyklusses mit hoher Wahrscheinlichkeit ($\geq \lambda$) entdeckt.

2.2 Merkle-Hashbäume und Chameleon-Hashing

Es seien die Daten (die Datei) $F = x_1 \| \ldots \| x_n$ gegeben. Eine Änderung an einem einzelnen Datensatz x_i der Datei erfordert i.d.R. das erneute datensatzweise Hashen von F, was im Allgemeinen $O(n)$ Aufrufe einer (kryptographischen) Hashfunktion H erfordert. Ein Merkle-Hashbaum ist eine Methode, um das Hashing so zu organisieren, dass eine derartige Änderung in $O(\log n)$ Aufrufen von H gelingt. Hierzu ordnet man die Blöcke von F als Blattknoten in einem binären Baum mit dem Wurzelknoten r^* an, und führt das Hashing rekursiv beginnend bei den Blättern durch. Der Hashwert eines Blattknotens x ist dabei festgelegt als $H(x)$, wobei x die im Knoten enthaltenen Daten repräsentiert. Der Hashwert eines inneren Knotens v mit den Kindknoten x, y ist definiert als $H(v) := H(H(x)\|H(y))$. Der Hashwert von F ist der Hashwert r^* des Wurzelknotens. Man beachte insbesondere, dass bei einer Änderung des Datensatzes $x_i \leftarrow \widetilde{x}_i$, eine Aktualisierung von r^* lediglich die Bekanntgabe von $O(\log n)$ vielen Hashwerten für die Geschwisterknoten entlang des Pfades vom Blatt x_i zur Wurzel (*Authentifizierungspfad*) erfordert.

Chameleon-Hashing: Eine *Chameleon-Hashfunktion* (auch *trapdoor commitment* genannt), ist eine (im Allgemeinen kollisionresistente) kryptographische Hashfunktion, für welche Kollisionen unter Zuhilfenahme eines geheimen Schlüssels effizient berechnet werden können. Wir geben hier nur die Definition einer solchen Hashfunktion wieder, und verweisen auf [AdM04] für Details und Sicherheitsbeweise. Eine Chameleon-Hashfunktion CH besteht aus drei Algorithmen:

`KeyGen`: Ein probabilitischer Algorithmus, welcher aus einem Sicherheitsparameter $t \in \mathbb{N}$ ein öffentliches/privates Schlüsselpaar (pk, sk) erzeugt.

`Hash`: Ein deterministischer Algorithmus, welcher aus den Eingabedaten x, dem öffentlichen Schlüssel pk und einer Zufallszahl r einen Hashwert $CH_{pk}(x, r)$ berechnet.

`Forge`: Ein deterministischer Algorithmus, welcher aus dem geheimen Schlüssel sk, einem Urbild (x, r) und dem zugehörigen Hashwert $CH_{pk}(x, r)$ eine Kollision (y, s) erzeugt für die gilt $CH_{pk}(x, r) = CH_{pk}(y, s)$.

Im Bezug auf Sicherheit benötigen wir im Folgenden nur die Kollisions-Resistenz von CH, im Sinne dass $\Pr[CH_{pk}(y, s) = CH_{pk}(x, r) | (y, s) \leftarrow \mathcal{A}(x, r, pk)]$ vernachlässigbar für alle (im Sicherheitsparameter) polynomiell beschränkten Angreifer $\mathcal{A}$ ist. Eine Beispielkonstruktion einer solchen Funktion wurde in [AdM04] angegeben:

`KeyGen`: Es seien p, q zwei große Primzahlen mit $p = uq + 1$, und g sei ein erzeugendes Element der Untergruppe der quadratischen Reste modulo p. Die Ordnung von g sei q. Setze $sk \in_R \{1, 2, \ldots, q-1\}$ und $pk \leftarrow g^{sk} \bmod p$. Wähle eine (herkömmliche) kryptographisch sichere Hashfunktion $H : \{0,1\}^* \to \{0,1\}^\ell$ mit $\ell \geq \lceil \log_2 p \rceil$.

`Hash`: Wähle $\rho, \delta \in_R \mathbb{Z}_p$, berechne $e \leftarrow H(m \| \rho)$ und definiere den Chameleon-Hashwert als $CH_{pk}(m, \rho, \delta) := (\rho - (pk^e g^\delta \bmod p)) \bmod q$.

`Forge`: Sei $C = CH_{pk}(m, \rho, \delta)$ gegeben. Wir wählen ein *beliebiges* $m' \neq m$ und eine Zufallszahl $k \in_R \{1, 2, \ldots, q-1\}$. Setze $\rho' \leftarrow (C + (g^k \bmod p)) \bmod q$, $e' \leftarrow H(m' \| \rho')$ und $\delta' \leftarrow (k - e' \cdot sk) \bmod q$. Die gesuchte Kollision tritt auf bei (m', ρ', δ'), zumal

$$\begin{aligned} CH_{pk}(m', \rho', \delta') &= \rho' - (pk^{e'} g^{\delta'} \bmod p) \bmod q \\ &= C + \left(g^k \bmod p\right) - \left(g^{sk \cdot e'} g^{\delta'} \bmod p\right) \bmod q \\ &= C = CH_{pk}(m, \rho, \delta). \end{aligned}$$

Zur Vereinfachung der Notation verzichten wir nachfolgend auf die explizite Angabe der Randomisierer ρ, δ und des öffentlichen Schlüssels pk, und schreiben kurz $CH(m)$ für $CH_{pk}(m, \rho, \delta)$. Man beachte hierbei insbesondere, dass der Teil m' des zweiten Urbildes (m', ρ', δ') *frei wählbar* ist. Diese Möglichkeit werden wir bei der Konstruktion unseres PoR ausnützen.

3 Konstruktion

Wir beschreiben die Komponenten des PoR-Protokolls einzeln, gemäß der in Abschnitt 2 dargestellten Struktur:

`Setup`: Initialisierung aller kryptographischen Parameter, insbesondere der Chameleon-Hashfunktion.

`Encode`: Wir nehmen an, dass die Datei $F = x_1 \| \ldots \| x_n$ bereits fehlerkorrigierend codiert ist. Es sei $S \subseteq \{1, 2, \ldots, n\}$ eine Teilmenge von Blöcken, welche wir stichprobenartig im Zuge des PoR überprüfen möchten. Wir erzeugen Paare der Form (c_i, r_i), wobei c_i ein zufälliger Bitstring (beliebiger Länge) ist, und r_i der Wurzel-Hash der mit c_i modifizierten Datei $F' = x_1 \| \ldots \| (c_i \| x_i) \| \ldots \| x_n$, d.h. wir konkatenieren c_i als Präfix des i-ten Datensatzes von F. Der Hashwert r_i wird gebildet durch Anwendung von CH in den Blattknoten, und Anwendung von H in den inneren Knoten (Merkle-Hashbaum). Man beachte, dass hierdurch die für CH verwendeten Randomisierer auf Seiten des Cloud-Provider geeignet codiert (etwa Base64) in die Blattknoten einzubetten sind. Der Client speichert den Wurzel-Hashwert r^* lokal.

`Challenge`: Wähle $i \in_R \{1, 2, \ldots, n\}$. Falls $i \in S$, sende (i, c_i) an den Server, andernfalls sende (i, c) mit einem Zufallsstring c.

`Response`: Der Server antwortet auf die Anfrage (i, c), indem er seinen lokal verwalteten Merkle-Hashbaum gemäß der Modifikation $x_i \leftarrow c \| x_i$ erneut erstellt (unter Verwendung von CH_{pk} in den Blattknoten). Er antwortet mit dem neuen Wurzelhashwert r', sowie dem Datum x_i einschließlich der zugehörigen Randomisierer (für CH). Diese Aktualisierung kann in $O(\log n)$ Aufrufen der Hashfunktion durchgeführt werden, und erfordert auch keinen zusätzlichen Speicherplatz, wenn der Merkle-Hashbaum in den zur Datei F gehörigen Index (Suchbaum) integriert wird (von dessen Existenz bei großen Datenmengen oder Datenbanken sinnvollerweise ausgegangen werden kann).

`Verify`: Die Antwort r' des Servers wird genau dann akzeptiert, wenn $(i \in S \wedge r_i = r') \vee (i \notin S)$. Die Menge S stellt hierbei die Menge bekannter Hashwerte dar, welche für die Verifikation der Existenz der Daten vorab beim Client berechnet und abgelegt wurde (anstelle der gesamten Datei F).

`Update`: Soll der Datensatz x_i durch $\widetilde{x}_i$ ersetzt werden, so fordert der Client den i-ten Datensatz an (per `Challenge`), und erhält (x_i, ρ_i, δ_i). Er konstruiert eine Hash-Kollision der Form $(\widetilde{x}_i, \rho_i', \delta_i')$ und sendet diese an den Server. Man beachte, dass aufgrund der unveränderten Hashwerte die Konsistenz sowohl der gesamten Daten am Server als auch der gesamten Verifikationsdaten des Clients erhalten bleibt.

`Extract`: Sequentielle Abfrage aller Datenblöcke und Verifikation des Wurzelhashwertes r^* gegen die erhaltenen Daten.

Für die Komplexitäten der einzelnen Operationen sei auf Anhang A, Tabelle 1 verwiesen.

4 Sicherheit

Anders als bei anderen PoR-Verfahren auf Basis von Stichprobenverifikationen erfolgt in dem hier beschriebenen System keine explizite Einbettung von Markern. Es entfallen somit die sonst üblichen Maßnahmen zur Verschleierung der Prüfdaten. Darüber steht dem Client die Möglichkeit offen, weitere Challenge-Response-Paare auf Basis des neuen Datensatzes $\widetilde{x}_i$ zu erzeugen. Da dieser Vorgang ohne Interaktion mit dem Server möglich ist, besitzt der Server keine Möglichkeit, diese neuen und implizit eingebetteten Marker zu entdecken. Gleichwohl muss der Server das Update durchführen (*freshness*-Eigenschaft), da spätere Anfragen genau dieses Datenwort betreffen könnten. Es ist somit eine quasi unbeschränkte Anzahl von Ausführungen des PoR möglich, welche gleichzeitig den Server zur Befolgung des Protokolls nötigen.

Der `Verify`-Algorithmus akzeptiert alle Antworten auf Anfragen betreffend Blöcke außerhalb von S. Dies dient der andauernden Verschleierung der Position von Prüfdaten im Zuge potentiell vieler Ausführungen des PoR, eröffnet jedoch gleichzeitig dem Server auch die Möglichkeit, die Position der Prüfdaten durch falsche Antworten quasi zu "testen". Die Anzahl der Prüfblöcke muss somit hinreichend groß gewählt werden, um dieser Attacke entgegenzuwirken. Der Beweis von Satz 4.1 ist in Appendix B angegeben.

Theorem 4.1 ([Ras13]) *Das beschriebene PoR-Verfahren ist* $(\rho, 1-|S|/n)$*-sicher gemäß Definition 2.1, wobei* ρ *vernachlässigbar im Sicherheitsparameter* t *ist.*

Insbesondere ist auch bei der Wahl der Hashfunktion H (auch als Baustein von CH) Vorsicht geboten: ein in [Ras13] nicht betrachteter Angriff nützt folgende Implementierungvariante aus, bei welcher die Zufallsdaten c als ***Suffix*** an das Datenwort x_i angehängt werden. In dieser Form erlauben standardisierte Hashfunktionen (wie MD5 [Riv92], SHA-1 [EJ01] und RIPEMD160 [DBP96]), eine Vorausberechnung des Hashwertes unter Verwendung aller Blöcke von x_i, bis zu dem Punkt, an dem die Daten c verarbeitet werden würden. Abbildung 1 zeigt das Prinzip einer iterativen Hashfunktion nach der Merkle-Damgård-Konstruktion [Mer89, Dam89] und dem bei MD5, SHA-1 und RIPEMD verwendeten Padding (Auffüllen des letzten 512 Bit Blocks mit $10\ldots0$ gefolgt von der Länge der zu hashenden Nachricht codiert als 64 Bit Wert). Da die Challenge c erst den Block b_{r-1} verändert, muss der Server in diesem Szenario nur das Zwischenergebnis h_{r-2} und den unvollständigen Block b_{r-1} speichern. Die übrigen Nutzdaten x_i – welche im Allgemeinen wesentlich mehr Speicherplatz beanspruchen – können verworfen werden. Die Werte $H(x_i\|c)$ können für beliebiges c jedoch weiterhin korrekt ermittelt werden.

Aus diesem Grund ist bei einer realen Implementierung, die Anfrage c stets als ***Präfix*** an das Datenwort zu konkatenieren. Inwieweit dieser Umstand zu Sicherheitslücken bei anderen Verfahren führen kann, welche ähnliche Konstruktionen verwenden (etwa auch die Funktion CH) ist eine aktuell offene Forschungsfrage. Für alternative Möglichkeiten um Hashfunktionen schlüsselabhängig zu machen, d.h. in sichere MACs umzuformen, sei hier außerdem auf [BCK96, Bel06] verwiesen. *Anmerkung:* Auch der neue Hash-Standard Keccak [BDPVA09] folgt in der "Absorptionsphase" ebenfalls dem Prinzip, wiederholt ein Zwischenergebnis mit Blöcken der zu hashenden Nachricht zu "mischen". D.h. gleiche Präfixe bei zu hashenden Nachrichten führen zu gleichen Zwischenergebnissen. Der

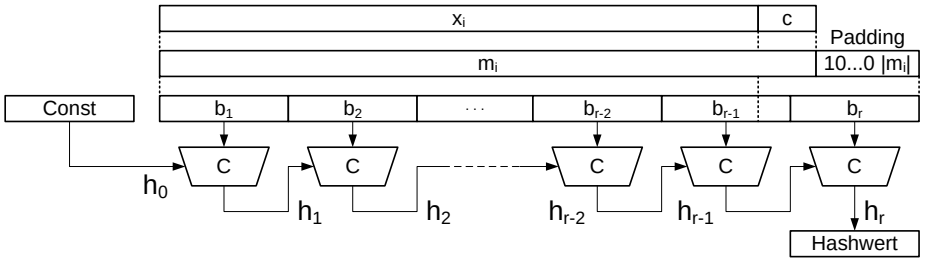


Abbildung 1: Hashen von $x_i \| c$: Merkle-Damgård-Konstruktion inkl. Padding

bei Keccak genutzte Sicherheitsparameter r, welcher die Anzahl der Nachrichten-Bits festlegt, die pro Iteration verarbeitet werden, könnte Teil der Challenge sein. Eine Vorausberechnung ist nun aber immer noch möglich, sie benötige nur mehr Speicherplatz (da die Ergebnisse der Vorausberechnungen für jeden Wert von r gespeichert werden müssen). Die Forderung nach *fairness* stellt Sicherheit für den Server her, im Sinne der Aufklärbarkeit von Anschuldigungen des Clients, dass die Daten modifiziert wurden. Auf technischer Ebene kann dies durch eine signierte Kette von Revisionen der Daten erreicht werden, welche ein gutes Cloud-Storage-System ohnedies durchführen sollte.

5 Resümee

Varianten und Erweiterungen: Eine Verallgemeinerung des Merkle-Hashbaumes von einer strikt binären Baumstruktur auf eine allgemeine Baumstruktur wie jene von XML-Dokumenten ist einfach. Da der Verzweigungsgrad des Hashbaumes für die Sicherheitsargumente keine Rolle spielt, bleiben die Sicherheitseigenschaften (insbesondere Theorem 4.1) gültig und erhalten. Allgemeinere Modifikationen wie das Einfügen oder Löschen von Datensätzen, oder auch strukturelle Änderungen der Daten bedürfen komplexerer Datenstrukturen als Merkle-Hashbäume. Dies ist Gegenstand aktueller Forschung. Im Hinblick auf den Schutz der Privatsphäre auch bei der Abfrage von Daten, kann sowohl eine Verschlüsselung der Datensätze vorgesehen werden, als auch eine Abfrage von wenigstens k Blöcken in jedem Schritt, von denen nur ein einziger wirklich von Interesse ist. In diesem Fall bleibt eine Rest-Unsicherheit von $\Theta(\log k)$ Bit für den Server, welcher aus k Datensätzen tatsächlich abgefragt wurde. Elegantere Methoden, welche den gleichen Effekt zum Ziel haben, bieten Techniken des *private information retrieval* [Gas04], auf welche wir hier nicht eingehen.

Ausblick: Dynamische PoR-Verfahren genießen aktuell großes Interesse seitens der wissenschaftlichen Gemeinschaft und stellen wichtige Bausteine für die Konstruktion sicherer Cloud-Storage-Dienste dar. Die vorliegende Arbeit zeigt eine einfache Konstruktion solcher dynamischen Verfahren auf Basis einfacher kryptographischer Standard-Bausteine. Praktische Implementierungen zum Zwecke der Messung von Effizienz und Aufwand sowohl für den Client als auch für den Server, sind Gegenstand laufender Betrachtungen.

Literatur

[ABC+07] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson und D. Song. Provable data possession at untrusted stores. In *CCS*, Seiten 598–609. ACM, 2007.

[AdM04] G. Ateniese und B. de Medeiros. On the key exposure problem in chameleon hashes. In *SCN*, Seiten 165–179. Springer, 2004.

[ADPMT08] G. Ateniese, R. Di Pietro, L. V. Mancini und G. Tsudik. Scalable and efficient provable data possession. In *SecureComm*, Seiten 9:1–9:10. ACM, 2008.

[AKK09] G. Ateniese, S. Kamara und J. Katz. Proofs of Storage from Homomorphic Identification Protocols. In *ASIACRYPT*, Seiten 319–333. Springer, 2009.

[BCK96] Mihir Bellare, Ran Canetti und Hugo Krawczyk. Keying Hash Functions for Message Authentication. In Neal Koblitz, Hrsg., *CRYPTO*, Jgg. 1109 of *LNCS*, Seiten 1–15. Springer, 1996.

[BDPVA09] G. Bertoni, J. Daemen, M. Peeters und G. Van Assche. Keccak specifications. online: http://keccak.noekeon.org/, 2009.

[Bel06] Mihir Bellare. New Proofs for NMAC and HMAC: Security Without Collision-Resistance. In Cynthia Dwork, Hrsg., *Advances in Cryptology – CRYPTO*, Jgg. 4117 of *Lecture Notes in Computer Science*, Seiten 602–619. Springer Berlin Heidelberg, 2006.

[BJO09] K. D. Bowers, A. Juels und A. Oprea. HAIL: a high-availability and integrity layer for cloud storage. In *CCS*, Seiten 187–198, 2009.

[CC12] B. Chen und R. Curtmola. Robust Dynamic Provable Data Possession. In *ICDCS Workshops*, Seiten 515–525. IEEE Computer Society, 2012.

[CKW12] D. Cash, A. Küpçü und D. Wichs. Dynamic Proofs of Retrievability via Oblivious RAM. In *IACR Cryptology ePrint Archive*, 2012. Report 2012/550.

[Dam89] I. Damgaard. A Design Principle for Hash Functions. In Gilles Brassard, Hrsg., *CRYPTO*, Jgg. 435 of *LNCS*, Seiten 416–427. Springer, 1989.

[DBP96] H. Dobbertin, A. Bosselaers und B. Preneel. RIPEMD-160: A Strengthened Version of RIPEMD. In D. Gollmann, Hrsg., *FSE*, Jgg. 1039 of *LNCS*, Seiten 71–82. Springer, 1996.

[DVW09] Y. Dodis, S. Vadhan und D. Wichs. Proofs of Retrievability via Hardness Amplification. In *TCC*, Seiten 109–127. Springer, 2009.

[EJ01] D. Eastlake und P. Jones. RFC RFC 3174: US Secure Hash Algorithm 1 (SHA1), 2001.

[EKPT09] C. Erway, A. Küpçü, C. Papamanthou und R. Tamassia. Dynamic provable data possession. In *CCS*, Seiten 213–222. ACM, 2009.

[Gas04] W. Gasarch. A Survey on Private Information Retrieval. *Bulletin of the EATCS*, 82:72–107, 2004.

[HHPSP11] S. Halevi, D. Harnik, B. Pinkas und A. Shulman-Peleg. Proofs of ownership in remote storage systems. In *CCS*, Seiten 491–500. ACM, 2011.

[JK07] A. Juels und B. Kaliski. PORs: Proofs of Retrievability for Large Files. In *CCS*, Seiten 584–597. ACM, 2007.

[LC11] S. Liu und K. Chen. Homomorphic Linear Authentication Schemes for Proofs of Retrievability. In *INCOS*, Seiten 258–262. IEEE Computer Society, 2011.

[LEB+03] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows und M. Isard. A cooperative internet backup scheme. In *USENIX ATEC*, Seiten 29–41. USENIX Association, 2003.

[Mer89] R. C. Merkle. One Way Hash Functions and DES. In G. Brassard, Hrsg., *CRYPTO*, Jgg. 435 of *LNCS*, Seiten 428–446. Springer, 1989.

[PSU12] M. Paterson, D. Stinson und J. Upadhyay. A coding theory foundation for the analysis of general unconditionally secure proof-of-retrievability schemes for cloud storage. *CoRR*, abs/1210.7756, 2012.

[Ras13] S. Rass. Dynamic Proofs of Retrievability from Chameleon-Hashes. In *SECRYPT '13*, Seiten 296–304. SciTePress, 2013.

[Riv92] R. Rivest. RFC 1321: The MD5 Message-Digest Algorithm, 1992.

[SW08] H. Shacham und B. Waters. Compact Proofs of Retrievability. In *ASIACRYPT*, Jgg. 5350 of *LNCS*, Seiten 90–107. Springer, 2008.

[WWR+11] Q. Wang, C. Wang, K. Ren, W. Lou und J. Li. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. *IEEE Trans. on Parallel and Distributed Systems*, 22(5):847–859, 2011.

[XC12] J. Xu und E.-C. Chang. Towards efficient proofs of retrievability. In *ASIACCS*, Seiten 79–80. ACM, 2012.

[ZX11] Q. Zheng und S. Xu. Fair and dynamic proofs of retrievability. In *CODASPY*, Seiten 237–248. ACM, 2011.

A Komplexitäten

Wir geben die asymptotischen Komplexitäten der einzelnen Verfahren hier in Abhängigkeit der Dateigröße n, gemessen in Blöcken, an. Hierbei wird eine Hash-Operation (egal ob Chameleon- oder herkömmlich) mit einem konstanten Aufwand $O(1)$ angenommen. Die vorgestellte Konstruktion ergibt die in Tabelle 1 angegebenen Komplexitäten [Ras13], wobei hier der Aufwand für fehlerkorrigierende Codierungen nicht berücksichtigt ist. Eine Implementierung im Rahmen derer Benchmarks durchgeführt und empirische Laufzeiten ermittelt werden ist Gegenstand laufender Aktivitäten.

	berechenmäßige Komplexität	
Operation	Verifier (Client)	Prover (Server)
Encode	$O(n \log n)$	$O(n \log n)$
Challenge	$O(1)$	–
Response	–	$O(\log n)$
Verify	$O(\log n)$	–
Update	$O(1)$	$O(1)$
neue Challenge	$O(\log n)$	–
Extract	$O(n \log n)$	$O(n \log n)$

Tabelle 1: Komplexitäten

B Beweis von Satz 4.1

Wir betrachten die Ereignisse

$$A := \left\{ \mathbf{Succ}^{\mathcal{A}}_{\mathsf{extract}}(F^*, \alpha) < 1 - \zeta \,\middle|\, [(F^*, \alpha) \leftarrow \mathbf{Exp}^{\mathcal{A}}_{\mathsf{setup}}(t)] \wedge [F \leftarrow \texttt{extract}^{\mathcal{A}_{\mathsf{resp}}}(\alpha)] \right\},$$
$$B := \left\{ \mathbf{Succ}^{\mathcal{A}}_{\mathsf{chal}}(F^*, \alpha) \geq \lambda \,\middle|\, [(F^*, \alpha) \leftarrow \mathbf{Exp}^{\mathcal{A}}_{\mathsf{setup}}(t)] \wedge [F \leftarrow \texttt{extract}^{\mathcal{A}_{\mathsf{resp}}}(\alpha)] \right\},$$

und zeigen $\Pr[\neg A \cup \neg B] \geq 1 - \zeta$. Hieraus folgt $\Pr[A \cap B] = \rho = \text{negl}(t)$, und damit die Sicherheit des Verfahrens gemäß Definition 2.1.

Das Ereignis $\neg A$ tritt genau dann ein, wenn der Verifizierer $F' = F^*$ mit überwältigender Wahrscheinlichkeit rekonstruiert. Nach Konstruktion überprüft `extract` ob $CH_{pk}(F^*) = r^*$ gilt, wobei r^* der beim Client vorab gespeicherte Wurzel-Hash der Original-Datei ist (welcher über eine beliebige Folge von Updates unverändert bleibt). Somit ist das Ereignis der irrtümlichen Akzeptanz äquivalent zum Ereignis einer Hash-Kollision $CH_{pk}(F') = CH_{pk}(F^*)$, dessen Wahrscheinlichkeit für eine kryptographisch sichere Hashfunktion als vernachlässigbar angenommen werden kann. Hieraus folgt $\Pr[\neg A] \geq 1 - \text{negl}(t)$.

Für das Ereignis $\neg B$ erinnern wir daran, dass der Angreifer mit Wahrscheinlichkeit von $\geq \lambda = 1 - |S| / |F|$ Fällen korrekt antworten kann (in diesen Fällen ist beim Client keine korrekte Antwort gespeichert, sodass der Client immer akzeptieren wird). In diesen Fällen gilt $\Pr[\neg B] = 0$.

Nach dem Satz von Sklår gilt $\Pr[\neg A \cap \neg B] = C(\Pr[\neq A], \Pr[\neg B])$ für eine Copula-Funktion C, welche die obere Frechet-Hoeffding-Ungleichung erfüllt, nämlich $C(x, y) \leq \min\{x, y\}$. Hieraus folgt $\Pr[\neg A \cap \neg B] \leq \min\{\Pr[\neg A], \Pr[\neg B]\} = 0$. Dies schließt den Beweis ab, da $\Pr[\neg A \cup \neg B] \geq 1 - \text{negl}(t) + 0 - 0$ und demzufolge $\Pr[A \cap B] = 1 - \Pr[\neg A \cup \neg B] \leq \text{negl}(t)$.

Verwendung von Festplattenvollverschlüsselung im geschäftlichen und privaten Umfeld

Christoph Sibinger Tilo Müller

Lehrstuhl für Informatik 1
Friedrich-Alexander-Universität Erlangen-Nürnberg
Martensstr. 3, 91054 Erlangen

christoph.m.sibinger@informatik.stud.uni-erlangen.de, tilo.mueller@informatik.uni-erlangen.de

Abstract: Festplattenvollverschlüsselung (engl. *Full Disk Encryption (FDE)*) stellt eine benutzerfreundliche und sichere Methode zum Schutz sensitiver Daten gegen physische Angriffe dar. Während es für den US-amerikanischen Markt in den letzten Jahren eine Reihe von Studien zur Verwendung von FDE gegeben hat, betrachtet die vorliegende Arbeit den deutschen Markt. Neben der Verwendung von FDE auf Laptops werden Smartphones in Betracht gezogen, und zusätzlich zum geschäftlichen Einsatz die private Nutzung untersucht. Zu diesem Zweck wurden Internet-gestützte Umfragebögen erstellt, in der die Teilnehmer zur eingesetzten Verschlüsselungstechnik und den jeweiligen Gründen befragt wurden, sowie Hintergrundwissen getestet wurde. Im Rahmen der Studie nahmen 1.034 Privatpersonen teil und 37 Unternehmen, wobei die Hälfte der befragten Unternehmen mindestens 1.000 Mitarbeiter beschäftigt und ein Drittel mindestens 10.000 Mitarbeiter.

1 Einleitung

Durch das Bundesdatenschutzgesetz (BDSG) schreibt der Gesetzgeber seit 1990 allen "Stellen, die selbst oder im Auftrag personenbezogene Daten erheben, verarbeiten oder nutzen" vor, "technische und organisatorische Maßnahmen zu treffen (...) um die Ausführung der Vorschriften dieses Gesetzes (...) zu gewährleisten". Diese Vorschrift gilt "wenn der Aufwand in einem angemessenen Verhältnis zu dem angestrebten Schutzzweck steht" (§9 BDSG). Durch eine zunehmend große Auswahl von Softwareprodukten zur Festplattenverschlüsselung, und der Verbreitung von Halbleiterlaufwerken (engl. *Solid-State Drives (SSDs)*) mit integrierter Hardware-Verschlüsselung, steht der oben genannte Aufwand heute i. d. R. in einem angemessenen Verhältnis zur Schutzbedürftigkeit der Daten. Neben diesen vom Gesetzgeber als schützenswert angesehenen Daten, gibt es in der Geschäftswelt schützenswerte Firmengeheimnisse, und auch einige Privatanwender haben ein Interesse daran ihre Daten abzusichern. In all diesen Szenarien bietet Festplattenverschlüsselung, egal ob hardware- oder softwarebasiert, einen relativ günstigen aber hohen Schutz gegen Datenlecks durch physischen Verlust oder Diebstahl von Datenträgern.

1.1 Motivation und Ziele

In welchem Umfang wird die Technologie der Festplattenverschlüsselung heute tatsächlich von Unternehmen eingesetzt? Nach welchen Gesichtspunkten werden die eingesetzten Lösungen ausgewählt? Sind sich Firmen und Privatpersonen auch der Schwachpunkte der von ihnen eingesetzten Lösung bewusst?

Während es für den Einsatz von Verschlüsselungstechnologien in amerikanischen Unternehmen bereits eine Reihe von Studien gibt (vgl. Kapitel 1.2), wird in der vorliegenden Arbeit erstmals der deutsche Markt betrachtet sowie Privatanwender hinzugezogen. Ziel dieser Arbeit ist die Erstellung und Auswertung eines Fragenkatalogs zum Einsatz von Festplattenvollverschlüsselungen bei in Deutschland angesiedelten Unternehmen und Anwendern.

1.2 Verwandte Arbeiten

Die im Jahr 2012 veröffentlichte Studie "US Full Disk Encryption 2011 Survey" [SEC12] der Firma SECUDE betrachtet 209 amerikanische Firmen. Die Studie kommt zu dem Ergebnis, dass in den USA 58% aller teilnehmenden Firmen im Jahr 2011 Festplattenverschlüsselungen einsetzte. Darüberhinaus gaben 25% an die Nutzung von Festplattenverschlüsselung zukünftig einführen oder ausbauen zu wollen. Allerdings gaben 25% der Betriebe an, dass – obwohl sie bisher keine Verschlüsselung einsetzen – auch in naher Zukunft keine Verschlüsselung eingeführt werden soll. Die Studie zeigt außerdem, dass ca. die Hälfte aller Firmen neben objektiven Anschaffungskriterien (wie bspw. dem Preis-/Leistungsverhältnis) viel Wert auf das Firmenimage des Partners legen. Wenig überraschend sind die Wünsche der Unternehmen hinsichtlich Benutzerfreundlichkeit und Performance: eine einfache, transparente Nutzung, die auch für unerfahrene Nutzer ohne Schulung möglich ist, und eine Systemleistung, die die tägliche Arbeit nicht beeinflusst, wird von mehr als drei Vierteln der Befragten als wichtig angesehen.

Die Studie "2010 Annual Study: US Enterprise Encryption Trends" [Pon10] des Ponemon Instituts wurde über ca. 1.000 Teilnehmer erhoben und ist der letzte Teil einer Serie von bis dahin jährlich durchgeführten Umfragen zu Verschlüsselungsstrategien amerikanischer Unternehmen. 59% der befragen Unternehmen setzten im Jahr 2010 eine Festplattenvollverschlüsselung ein. Trotzdem beklagten in diesem Zeitraum zwei Drittel der Unternehmen mehrere Vorfälle eines Datenlecks; lediglich 12% gaben an, nie ein derartiges Vorkommen gehabt zu haben. Außerdem zeigt die Studie, dass Unternehmen Verschlüsselung inzwischen weniger zur freiwilligen Abwehr von Datenlecks betreiben, als vielmehr zur Erfüllung von Datenschutzrichtlinien durch den Gesetzgeber.

1.3 Übersicht der Ergebnisse

Wie sich im Rahmen unserer Studie herausgestellt hat, ist Verschlüsselung im geschäftlichen Umfeld in Deutschland weit verbreitet. 85% aller befragten Unternehmen gaben an ihre

Festplatten zu verschlüsseln, so dass FDE durchaus als Standard bei deutschen Unternehmen gesehen werden kann. Hardwarebasierte Verschlüsselung wird noch verhältnismäßig selten eingesetzt (20%), während der Großteil der Firmen per Software verschlüsseln, in erster Linie mit Microsoft BitLocker. Die Verschlüsselung von Mitarbeiter-Smartphones ist etwas weniger verbreitet (68%), was allerdings auch darauf zurückzuführen ist, dass einige Firmen die Speicherung sensitiver Daten auf diesen Geräten nicht erlauben. Die wichtigsten Kriterien deutscher Firmen für den Erwerb einer Verschlüsselungslösung sind die Usability (62%), die Performance (51%), eine einfache Inbetriebnahme (51%), der Preis (46%) und das Image des Herstellers (41%).

Im Gegensatz zu Unternehmen liegt der Anteil der untersuchten Privatanwender, die Festplattenverschlüsselung einsetzen, mit ca. 25% deutlich niedriger. Zudem ist unsere Umfrage bei Privatanwendern nicht repräsentativ für die deutsche Allgemeinheit, sondern umfasst vor allem Studenten und Angestellte der Technischen und der Wirtschaftswissenschaftlichen Fakultät der Universität Erlangen-Nürnberg. Es ist davon auszugehen, dass der Anteil der Privatanwender, die Festplattenverschlüsselung einsetzen, gemessen an der gesamtdeutschen Bevölkerung weniger als 25% beträgt. Der Grund ist eine signifikante Korrelation zwischen dem Einsatz von Verschlüsselung und PC-Kenntnissen, die wir feststellen konnten (Cramers $V = 0,398$, $p < 0,001$). Interessanterweise wird von den befragten Teilnehmern die Sicherheit hardwarebasierter Verschlüsselung im Schnitt schwächer bewertet als die Sicherheit softwarebasierter Methoden. Die Verschlüsselung von Smartphones ist relativ weit verbreitetet und war bei immerhin 42% der untersuchten Teilnehmer aktiviert.

2 Technischer Hintergrund

Im Folgenden wird kurz auf die verschiedenen Ausprägungen von Festplattenverschlüsselung (Kapitel 2.1) eingegangen, sowie auf deren Sicherheit (Kapitel 2.2). Insbesondere werden Schwachstellen aufgezeigt und einige, in der Praxis relevante Angriffe auf Verschlüsselungssysteme dargestellt.

2.1 Arten der Festplattenverschlüsselung

Eine *Festplattenverschlüsselung*, oder *Festplattenvollverschlüsselung* (engl. *Full Disk Encryption (FDE)*), zeichnet sich gegenüber der Verschlüsselung einzelner Dateien und Verzeichnisse dadurch aus, dass ganze Partitionen einer Platte verschlüsselt werden. Für den Anwender, wie auch die Anwendersoftware, ist die Verschlüsselung damit transparent, denn es wird unterhalb der Dateisystemebene verschlüsselt. Dies kann technisch entweder im Betriebssystemkern erfolgen, in diesem Fall sprechen wir von *softwarebasierter* Verschlüsselung, oder durch die Festplatte selbst, in dem Fall sprechen wir von *hardwarebasierter* Verschlüsselung.

2.1.1 Softwarebasierte Verschlüsselung

Bei softwarebasiertem FDE werden alle Daten durch den Betriebssystemkern verschlüsselt bevor sie auf die Festplatte geschrieben werden, bzw. entschlüsselt nachdem sie gelesen wurden. Für den Ver- und Entschlüsselungsschritt wird jeweils der Hauptprozessor und -speicher beansprucht. Da von derart gesicherten Medien nicht gestartet werden kann, existiert zusätzlich ein unverschlüsselter Bootloader, der vor dem Starten des eigentlichen Systems den Benutzer authentifiziert und das notwendige Passwort abfragt. Zu den bekanntesten softwarebasierten Verfahren zählen Microsoft BitLocker und die Open-Source Lösung TrueCrypt. Unter Apple MacOS findet darüberhinaus FileVault Verwendung, und unter Linux meist dm-crypt. Android-basierte Smartphones verfügen seit Version 4.0 ebenfalls über eine Option die Benutzer-Partition zu verschlüsseln (jedoch nicht die System-Partition), und iPhones verfügen ebenfalls über eine Verschlüsselung.

Solche softwarebasierten Verfahren haben zwei Nachteile: einerseits sinkt durch die Belastung des Hauptprozessors die Performance des Gesamtsystems, und andererseits eröffnet die Nutzung des Hauptspeichers Angriffsmöglichkeiten wie Cold-Boot (vgl. Kapitel 2.2). Dennoch haben unsere Untersuchungen gezeigt, dass softwarebasiertes FDE, vor allem BitLocker und TrueCrypt, heute sehr beliebt sind. Dies kann historisch erklärt werden, denn softwarebasierte Methoden existieren seit etwa 10 Jahren, und damit wesentlich länger als hardwarebasierte Methoden. Außerdem sind softwarebasierte Methoden flexibler, unabhängig von bestimmten Hardware-Herstellern, und teilweise kostenlos.

2.1.2 Hardwarebasierte Verschlüsselung

Seit etwa 4 Jahren finden sogenannte *selbstverschlüsselnde Festplatten* (engl. *Self Encrypting Drives (SEDs)*) zunehmend Verbreitung. Diese führen die Ver- bzw. Entschlüsselung transparent für das Betriebssystem in Hardware durch und basieren i.d.R. auf modernen *Solid-State Drives (SSDs)*, da diese einen komplexen Disk-Controller aufweisen der relativ einfach um die Funktion einer Verschlüsselung ergänzt werden kann. Beispiele für SEDs sind die Fujitsu DX8090, Seagate Cheetah, Intel 320 oder 520 und Samsung 830.

Im Unterschied zu softwarebasierter Verschlüsselung können SEDs stets alle Daten einer Festplatte verschlüsseln. Ein unverschlüsselter Bootloader ist nicht nötig, weil das BIOS die Aufgabe der Passwortabfrage übernimmt und die Festplatte entsperrt. Nach der Entsperrung verschlüsseln sich SEDs dann autonom und verhalten sich nach außen wie unverschlüsselte Platten. Die Performance des Hauptsystems wird daher nicht beeinflusst und prinzipiell werden alle Betriebssysteme unterstützt, auch wenn sie selbst keine Verschlüsselungsoption bieten. Nachteilig sind unter Umständen höhere Anschaffungskosten und der vergleichsweise geringe Speicherplatz auf SSDs zu nennen.

2.2 Sicherheit von Festplattenverschlüsselung

Wie unsere Untersuchungen zeigen, genießt softwarebasiertes FDE in Deutschland sowohl eine höhere Verbreitung als auch ein größeres Vertrauen unter den Anwendern. Das dieses Vertrauen nicht in jedem Fall gerechtfertigt ist, wollen wir kurz anhand einer Reihe von bekannten Schwachstellen zeigen.

Cold-Boot Angriffe Cold-Boot Angriffe greifen die Schlüsselverwaltung im Hauptspeicher an und richten sich somit gegen softwarebasierte Verschlüsselung. Entgegen der Annahme vieler Menschen, verflüchtigen sich Inhalte des Hauptspeichers nicht sofort nach dem Ausschalten eines Rechners, sondern bleiben mehrere Sekunden ohne Strom erhalten. Im Jahr 2008 konnten Halderman et al. [HSH+08] erstmals zeigen, dass sich dieser Effekt für praktische Angriffe gegen Festplattenverschlüsselungen ausnutzen lässt. Unter der Voraussetzung, dass ein Zielsystem eingeschaltet ist (oder sich im Bereitsschaftszustand befindet), entnimmt der Angreifer die Speicherriegel, baut diese in einen Analyserechner ein und durchsucht sie nach dem Festplattenschlüssel. Auf diese Weise konnten BitLocker und TrueCrypt erfolgreich angegriffen werden. 2013 zeigten Müller und Spreitzenbarth [MS13] darüberhinaus, dass ähnliche Angriffe auch gegen die Verschlüsselung von Android eingesetzt werden können.

Evil-Maid Angriffe Dieser Angriff, welcher im Wesentlichen das Benutzerpasswort erspäht, richtet sich ebenfalls gegen softwarebasierte Festplattenverschlüsselung. 2009 beschrieb Rutkowska [Rut09] folgendes, namensgebende Angriffsszenario gegen TrueCrypt: Ein Hotelgast lässt seinen ausgeschalteten Laptop unbeaufsichtigt zurück, so dass ein "böses Zimmermädchen" unbemerkt Zugriff auf diesen erlangen kann. Für das Zimmermädchen ist es nun möglich den unverschlüsselten Teil des Bootloaders mit einem Keylogger zu infizieren. Sobald der Hotelgast zurückkehrt und sich anmeldet, wird das Benutzerpasswort auf der Festplatte geloggt. Mit einem zweiten Zugriff kann das Zimmermädchen dieses auslesen und somit die Festplattenverschlüsselung umgehen. 2013 zeigten Götzfried und Müller [GM13], dass der Angriff prinzipiell auch gegen Android funktioniert.

Direct-Memory-Access Angriffe DMA Angriffe richten sich gegen software- und hardwarebasierte Verschlüsselung gleichermaßen. 2005 haben Dornseif et al. [DBK05] erstmals gezeigt, dass sich DMA-fähige Schnittstellen wie FireWire ausnutzen lassen um eingeschaltete Rechner zu entsperren. Der Grund ist, dass per DMA angeschlossene Geräte vollen Lese- und Schreibzugriff auf den Hauptspeicher eines Systems haben. Durch Schreibzugriff auf den Systemspeicher lässt sich eine Bildschirmsperre nun ohne weiteres aushebeln, obwohl das Passwort unbekannt ist. Ähnlich zu Cold-Boot Angriffen muss das Zielsystem dabei eingeschaltet sein (oder sich im Bereitschaftszustand befinden). Obwohl Gegenmaßen seit Jahren existieren, sind alle Windows-Versionen bis einschließlich Version 8 von dieser Schwachstelle betroffen.

Hot-Plug Angriffe 2012 stellten Müller, Latzo und Freiling [MLF13] erstmals einen Angriff vor, der sich ausdrücklich gegen hardwarebasierte Verschlüsselung richtet; softwarebasierte Verschlüsselung ist nicht betroffen. Grundlage dieses Angriffs ist, dass die Strom- und Datenkabel von SEDs getrennt geführt werden, SEDs aber nur gesperrt werden sobald das Stromkabel getrennt wird. Das Datenkabel kann von einem Zielrechner abgeklemmt und an einen Analyse-Rechner angeschlossen werden, ohne dass SEDs dabei gesperrt werden. Wenn der Zielrechner eingeschaltet ist (oder sich im Bereitschaftszustand befindet), erlangt ein Angriffer dadurch vollen Zugriff auf die Daten und umgeht somit die Verschlüsselung.

Zusammenfassend ist zu sagen, dass Festplattenverschlüsselung, gleich welcher Art, wesentlich sicherer ist wenn ein System vollständig heruntergefahren ist. Unternehmen sollten ihre Mitarbeiter dazu anhalten Systeme niemals eingeschaltet (oder im Bereitsschaftszustand befindlich) unbeaufsichtigt zu lassen, sondern stets herunterzufahren.

3 Studie zur Verwendung von Festplattenvollverschlüsselung

Im Folgenden präsentieren wir die Umfrageergebnisse unserer Studie zur Verwendung von FDE. Die Kernfragen unserer Studie zum privaten Umfeld (Kapitel 3.1) und der zum geschäftlichen Umfeld (Kapitel 3.2) sind sich ähnlich. Beide Fragebögen unterscheiden sich im Wesentlichen dadurch, dass sich der private Fragebogen auf einzelne Laptops bzw. Smartphones bezieht, während im professionellen Bereich nach dem Gesamtbestand eines Unternehmens gefragt war.

3.1 Festplattenverschlüsselung im privaten Umfeld

Der Großteil der privaten Antworten ging über eine Onlineumfrage ein, während einzelne analoge Antworten nachträglich in das System eingepflogen wurden. Am Ende gab es 1379 Datensätze von denen, nach Aussortierung unvollständiger und widersprüchlicher Antworten, 1034 ausgewertet wurden. Um Abbrüche und Antworten nach dem Zufallsprinzip zu vermeiden, waren die Teilnehmer nicht gezwungen alle Fragen zu beantworten. Um möglichst viele Privatanwender zu erreichen, wurde die Studie neben Facebook über verschiedene Mailinglisten der Universität Erlangen-Nürnberg angekündigt. Durch die Zielgruppe der Mailinglisten ist die Umfrage nicht repräsentativ für die Allgemeinheit, sondern umfasst vor allem Studenten und Angestellte der Technischen und der Wirtschaftswissenschaftlichen Fakultät. Dies erklärt den hohen Anteil junger und männlicher Probanden (vgl. Tabelle 1). Die PC-Kenntnisse der Teilnehmer (vgl. Tabelle 2) liegen womöglich deutlich über dem Bevölkerungsdurchschnitt.

90% der Befragten gaben an einen Laptop zu besitzen. Den Markt teilen sich dabei die Hersteller wie folgt auf: Acer (15%), Lenovo (14%), Apple (12%) und Dell (11%), Asus, HP und Samsung (je unter 10%) und schließlich sonstige Hersteller (je unter 5%). Der große Vorteil eines Laptops – seine Mobilität – wird von fast allen Befragten genutzt; lediglich

männlich	69%
weiblich	28%
keine Angabe	3%
18-24	64%
25-34	32%
sonstiges Alter	4%

Tabelle 1: Geschlechts- und Alterstruktur.

sehr gut (1)	22%
gut (2)	28%
durchschnittlich (3)	30%
schlecht (4)	8%
sehr schlecht (5)	2%
Durchschnitt	2,29

Tabelle 2: PC-Kenntnisse der Teilnehmer.

8% gaben an ihren Laptop nur zu Hause zu benutzen. Die Verlustquote bei privaten Laptops liegt bei 4%. Smartphones sind nicht so verbreitet wie Laptops; nur 2/3 der Befragten gaben an ein Smartphone zu besitzen. Die Marktanteile sind dabei konzentrierter als bei Laptops: Samsung (37%), Apple (20%) und HTC (16%) teilen sich über 70% des Marktes. Sony und Nokia folgen mit 7% respektive 5%. Die höhere Verlustrate von 9% bei Smartphones erstaunt nicht.

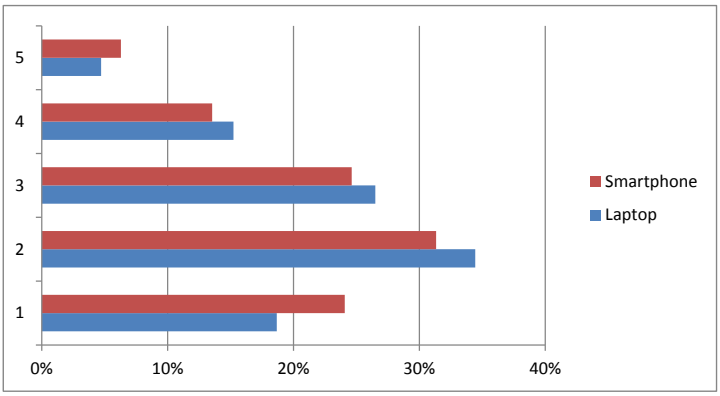


Abbildung 1: Potentieller Schaden durch Datendiebstahl (von 1: kein Schaden bis 5: hoher Schaden).

Die Höhe des potentiellen Schadens, den ein Diebstahl der gespeicherten Daten auf einem Laptop bzw. Smartphone für einen Probanden zur Folge hat, zeigt Abbildung 1. Dabei handelt es sich um den subjektiven Eindruck eines Teilnehmers auf einer Skala von 1 (kein Schaden) bis 5 (hoher Schaden). Der durchschnittliche Schaden wird mit 2,53 (Laptops) und 2,47 (Smartphones) für beide Geräteklassen ähnlich angegeben.

3.1.1 Eingesetzte Verschlüsselung

Abbildung 2 verdeutlicht, dass die Mehrheit der privaten Nutzer keine Verschlüsselung auf Laptops einsetzt (über 70%). Wenn verschlüsselt wird, so wird i.d.R. auf softwarebasierte Verschlüsselung zurückgegriffen (19%). Der Markt der softwarebasierten Verschlüsselung wird dabei von TrueCrypt dominiert (63%), gefolgt von dm-crypt für Linux (16%), FileVault für MacOS (7%) und BitLocker für Windows (4%). Selbstverschlüsselnde Festplatten (SEDs) nutzen nur etwa 4% der Befragten, wobei ein Großteil dabei interessanterweise

angibt zusätzlich softwarebasierte Verschlüsselung zu nutzen. Technisch ist dies zwar möglich, bedeutet in der Praxis aber kaum einen Sicherheitsgewinn.

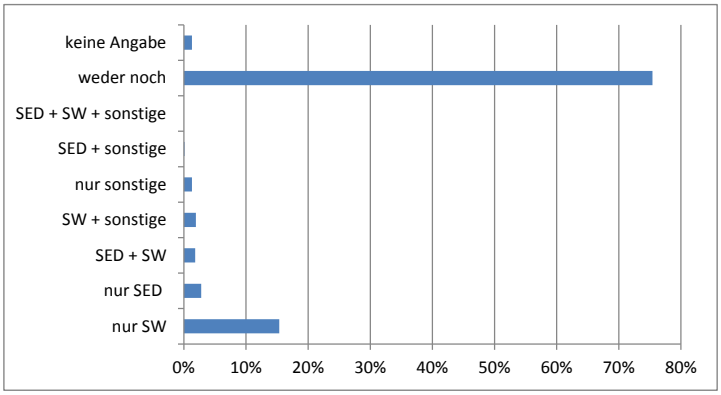


Abbildung 2: Verwendete Verschlüsselungsprodukte für Laptops bei Privatpersonen.

Im Smartphonebereich ist die Verschlüsselungsquote deutlich höher als bei Laptops, obwohl das Schadenspotential der Daten als vergleichbar angegeben wird. 42% der Befragten haben ihr Smartphone verschlüsselt. Eine Erklärung dafür ist, dass viele Smartphones heute eine einfach zu aktivierende Verschlüsselung bieten (Android) oder gar ab Werk verschlüsselt sind (iPhone), was bei vielen Laptops nicht der Fall ist. So ist BitLocker bspw. nur in den Enterprise-Versionen von Microsoft Windows verfügbar.

Eine Korrelationsanalyse zwischen PC-Kenntnissen und dem Einsatz von Verschlüsselung auf Laptops ergibt einen mittleren Zusammenhang (Cramers $V = 0,398$, $p < 0,001$). Dies ist insofern nicht überraschend, als dass davon auszugehen ist, dass es vor allem erfahrene Nutzer sind, die ihre Systeme mit Funktionen schützen, die nicht standardmäßig installiert bzw. aktiviert sind. Eine ähnliche Korrelationsanalyse für den Einsatz von Verschlüsselung auf Smartphones ergibt einen geringeren Zusammenhang (Cramers $V = 0,118$, $p < 0,1$).

Codestellen	numerisch	alphanumerisch	keine Angabe	Summe
4	201	11	0	212
5	29	2	1	32
6	28	4	0	32
7	7	2	0	9
8	22	19	0	41
9	10	5	0	15
10	3	8	0	11
11	0	2	2	4
12	5	26	2	33
mehr	0	3	0	3

Tabelle 3: Art und Länge von Code-Sperren bei privaten Smartphones.

Bei Diskussionen über die Sicherheit von Verschlüsselung wird immer wieder angeführt, dass diese nur so sicher ist wie das gewählte Passwort. Aus diesem Grund wurde in unserer

Umfrage nach der Länge und Art der Code-Sperren bei Smartphones gefragt. Inbesondere bei Smartphones ist man aus Gründen der Benutzerfreundlichkeit dazu verleitet kurze PINs bzw. Passwörter zu wählen. Tabelle 3 zeigt, dass die Codestärke der meisten Nutzer in der Tat relativ schwach ist. Eine Häufung der Passwortlänge von 4 ist dadurch zu erklären, dass dies bei vielen Geräten die Mindestlänge darstellt. Geht man davon aus, dass sichere Passwörter mindestens 8 alphanumerische Zeichen haben, so benutzen nur 9% der Anwender ein sicheres Passwort.

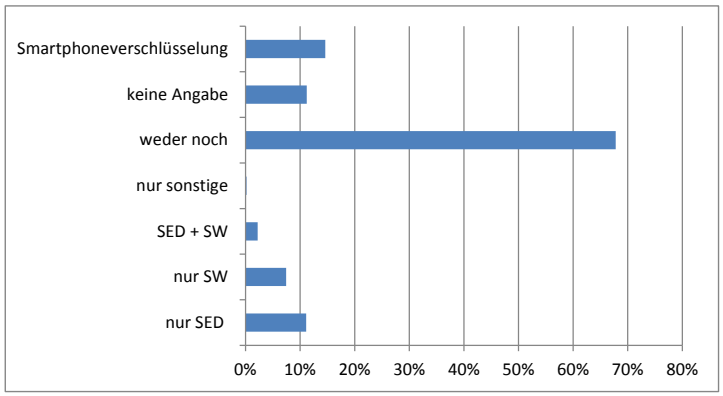


Abbildung 3: Geplanter Einsatz von Verschlüsselung im privaten Umfeld.

Außerdem wurde gefragt, inwieweit die Teilnehmer planen zukünftig Verschlüsselungsprodukte einzusetzen. Weitere 19% der Befragten gaben an, in Zukunft ihre Daten auf dem Laptop verschlüsseln zu wollen. Der Anteil von SEDs gemessen am Gesamtaufkommen von Verschlüsselungslösungen wird dabei steigen (vgl. Abbildung 3).

3.1.2 Entscheidungsgründe und Einschätzungen

In einer Freitextantwort konnten die Teilnehmer ihre Gründe für bzw. gegen den Einsatz von Verschlüsselungsprodukten angeben, wovon 462 Teilnehmer (45%) Gebrauch machten. Die Ergebnisse dazu zeigt Tabelle 4. Hauptgrund für die Verschlüsselung ist erwartungsgemäß die Verbesserung der Datensicherheit, während andere Gründe nur vereinzelt genannt wurden. Die Argumente gegen die Verschlüsselung sind vielfältiger. Der Verzicht auf Verschlüsselung bei Abwesenheit schützenswerter Daten ist nachvollziehbar. Zudem ist Unwissenheit häufig genannt und die Tatsache, dass sich viele Nutzer vor dem Aufwand einer Neuinstallation scheuen.

Schließlich wurden die Teilnehmer gebeten die Sicherheit von hardware- und softwarebasierter Verschlüsselung einzuschätzen (Abbildung 4) bzw. anzugeben wo sie Vor- und Nachteile dieser Technologien sehen (Abbildung 5). Dabei nehmen wir aufgrund der hohen Anzahl derjenigen an, die keine Angaben gemacht haben, dass viele Befragte zu wenig technisches Verständnis von Festplattenverschlüsselung hatten, um diese Fragen sinnvoll zu beantworten.

Gründe gegen Verschlüsselung		Gründe für Verschlüsselung	
keine Notwendigkeit	175	Datensicherheit	105
Unwissenheit	87	einfache Installation	11
hoher Aufwand	68	Interesse	4
kein Diebstahlrisiko	41	Paranoia	3
kein Interesse	28	günstiger Preis	3
Performanceeinbußen	21	externe Vorgaben	1
Angst vor Datenverlust	21	SED mitgeliefert	1
kein Vertrauen in die Sicherheit	8	gute Performance	1
keine (gute) OS Unterstützung	5		

Tabelle 4: Gründe für bzw. gegen den Einsatz von Verschlüsselung im privaten Umfeld.

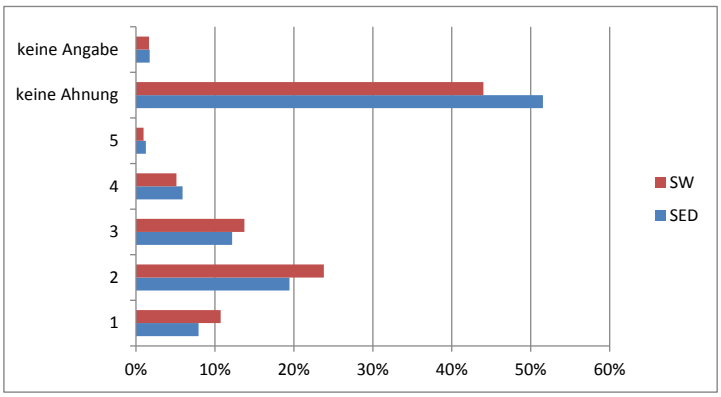


Abbildung 4: Einschätzung der Sicherheit (von 1: sehr gut bis 5: sehr schlecht).

Die Sicherheit von SEDs wird von den Teilnehmern im Schnitt schwächer bewertet als die softwarebasierter Methoden (SEDs: 2,42 vs SW: 2,30). Aufgrund verfügbarer Open-Source Lösungen ist es wenig überraschend, dass beim Preis ebenfalls die Vorteile bei softwarebasierter Verschlüsselung genannt werden. Weitere Eigenschaften ergaben leichte Vorteile zugunsten von SEDs, inbesondere die Usability, Performance und Installation.

Als letztes wurden die Teilnehmer gefragt, ob sie die in Kapitel 2.2 beschriebenen Angriffe kennen. Der bekannteste Angriff war der DMA bzw. FireWire Angriff mit 23%, gefolgt von Cold-Boot und Hot-Plug mit je 15%, und schließlich dem Evil-Maid Angriff mit 7%.

3.2 Festplattenverschlüsselung im geschäftlichen Umfeld

Von 59 geschäftlichen Datensätzen, die ebenfalls meist online eingegangen sind, wurden am Ende 37 vollständige und widerspruchsfreie Datensätze ausgewertet. Die Kontaktaufnahme zu potentiellen Teilnehmern erfolgte auf drei verschiedene Wege: durch persönliche Kontakte, durch das Kompetenzforum Cyber-Sicherheit Deutschland und durch offizielle

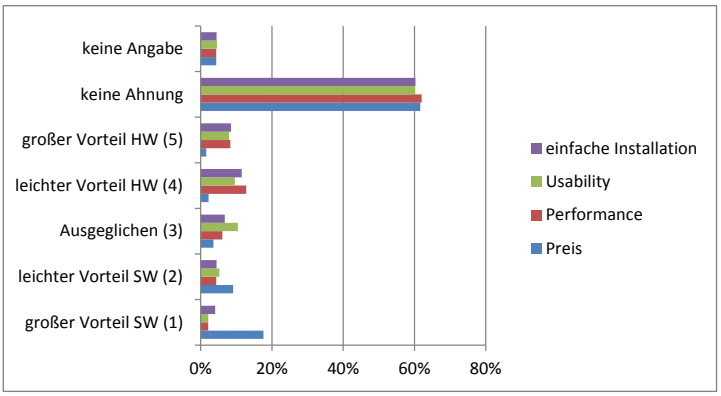


Abbildung 5: Vergleich von hardware- (HW) und softwarebasierter (SW) Verschlüsselung.

Kontaktadressen der Firmen. Neben Unternehmen wurde auch Ämter, Ministerien, Polizeibehörden, Forschungseinrichtungen und andere öffentliche und halböffentliche Institutionen angeschrieben.

Lehre und Wissenschaft	4
Behörde	2
Finanzdienstleister	2
Industrie und Einzelhandel	6
Dienstleister	6
Gesundheits- und Pharmaindustrie	3
Technologie- und Softwareunternehmen	3
Telekommunikationsunternehmen	3
Konsumgüter und Transportwesen	2
sonstige	6

Tabelle 5: Branchen der teilnehmenden Unternehmen.

keine Angabe	5	8
1-5	0	0
6-20	4	6
21-100	8	9
101-500	6	1
501-1000	1	4
1001-5000	5	5
5001-10000	5	2
10001+	3	2

Tabelle 6: Im Umlauf befindliche Laptops (li.) und Smartphones (re.).

Innerhalb der Stichprobe von 37 Unternehmen konnte ein breites Spektrum an Branchen abgedeckt werden (vgl. Tabelle 5). Außerdem war es möglich relativ viele Konzerne mit großen Mitarbeiterzahlen und einer entsprechend großen Anzahl firmeneigener Laptops und Smartphones für die Teilnahme zu gewinnen. Fast die Hälfte der befragten Unternehmen hat mindestens 1.000 Beschäftigte, 2/3 davon mindestens 10.000. 34 Firmen geben dabei Laptops zur täglichen Arbeit an ihre Mitarbeiter aus.

Im geschäftlichen Bereich sind die meistgenutzten Laptophersteller Dell und Lenovo (je über 40%) sowie HP (24%), gefolgt von Apple (11%) und Fujitsu (16%). Die Verlustquote für Laptops wird mit maximal 4% angegeben, ist mit durchschnittlich 0,7% aber sehr gering und steht damit im Gegensatz zu den uns bekannten amerikanischen Studien [Pon10, SEC12]. Möglicherweise liegt die Dunkelziffer hier um einiges höher; ein Teilnehmer gab an, dass es im Konzern keine Meldepflicht für verlorene Laptops gäbe.

Smartphones sind in Unternehmen ähnlich verbreitet wie Laptops (vgl. Tabelle 6). 33

der befragten Betriebe geben Smartphones an ihre Mitarbeiter aus. Die meistgenutzten Hersteller (Mehrfachnennung möglich) sind dabei Apple und Research in Motion (je knapp 60%), Samsung (33%), HTC (24%), Nokia (12%) und Sony (3%). Die Verlustquote ist mit 1,1% im Schnitt etwas höher als bei Laptops, und entspricht etwa 25 verlorenen Smartphones pro Firma und Jahr.

3.2.1 Datensensitivität

Inwieweit eine Verschlüsselung Sinn macht, hängt maßgeblich von der Sensitivität der Daten ab. Deshalb wurde gefragt auf wievielen Laptops bzw. Smartphones vertrauliche Daten gespeichert werden (vgl. Abbildung 6). Interessant ist dabei die unterschiedliche Verteilung für Smartphones und Laptops: während die Verteilung für Laptops grob einer Normalverteilung entspricht, werden sensitive Daten auf Smartphones entweder möglichst vermieden oder vergleichsweise häufig gespeichert. Dabei unterliegen die Daten sowohl auf Laptops als auch auf Smartphones häufig Gesetzen wie dem BDSG. 71% aller Unternehmen gaben an, BDSG Daten auf Laptops zu speichern, und 59% aller Unternehmen gaben an, dies ebenfalls auf Smartphones zu tun. Dementsprechend hoch ist der potentielle Schaden, den ein Diebstahl der gespeicherten Daten verursachen kann. Mehr als ein Drittel erwarten im Ernstfall hohen bzw. sehr hohen Schaden durch Datendiebstahl.

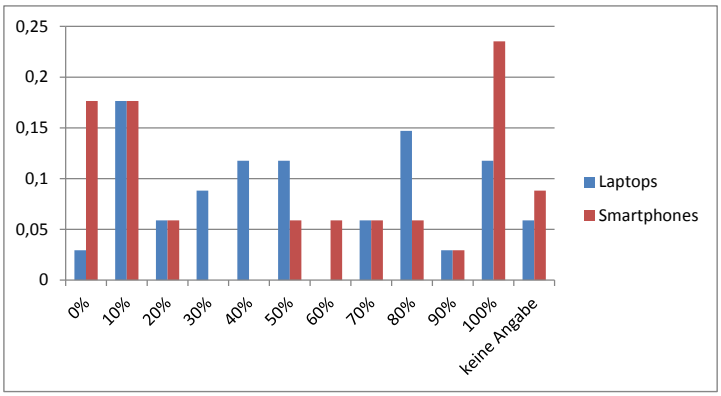


Abbildung 6: Hardware mit vertraulichen Daten im geschäftlichen Umfeld. Die Y-Achse beschreibt den Anteil der Firmen die auf dem durch X angegeben Anteil ihrer Hardware kritische Daten speichert.

3.2.2 Eingesetzte Verschlüsselung

Es zeigt sich, dass der Anteil von Unternehmen die Verschlüsselung einsetzen mit fast 85% im Vergleich zu Privatanwendern relativ hoch ist. Lediglich 15% aller Unternehmen setzen keine Produkte zur Festplattenverschlüsselung ein (vgl. Abbildung 7).

Wie im privaten Gebrauch, sind SEDs auch im professionellen Einsatz vergleichsweise selten anzutreffen. Da die meisten Unternehmen eine große, gewachsene IT-Struktur haben, ist ein kompletter Umstieg auf SEDs oftmals nur schwer zu realisieren. Deshalb geben

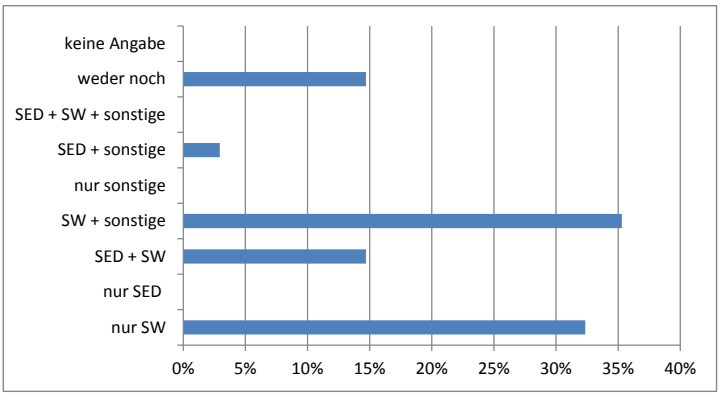


Abbildung 7: Verwendete Verschlüsselungsprodukte in Unternehmen.

alle Firmen, die bereits SEDs nutzen, zusätzlich auch an softwarebasierte Produkte zu verwenden. Dies ist aber nicht, oder nicht unbedingt, gleichbedeutend mit der Praxis beide Technologien auf ein und demselben Laptop einzusetzen. In 50% der Unternehmen die SEDs einsetzen, werden lediglich 10% der Laptops tatsächlich auf diese Weise geschützt. Zwei Firmen die bisher softwarebasierte Produkte verwenden, planen in naher Zukunft den Einsatz von SEDs. Eine Firma, die ihre Daten bisher nicht schützt, plant dagegen die Nutzung einer softwarebasierten Lösung einzuführen.

Im Softwaresegment ist Microsoft BitLocker, das nur 4% der Privatanwender einsetzen, mit Abstand das meistgenutzte Produkt. Über 50% der Firmen geben an, BitLocker im Einsatz zu haben. Darauf folgen Sophos Safeguard (18%), SECUDE Finally Secure (11%), McAfee Endpoint Encryption (11%), PGP Whole Disk Encryption (7%), Check Point PointSec (7%) und Apple FileVault (7%).

Betrachtet man wieviele Laptops innerhalb eines Unternehmens verschlüsselt werden, so geben 69% der Unternehmen an alle Laptops (100%) zu verschlüsseln. Weitere 10% der Unternehmen verschlüsseln mindestens 90% ihrer Laptops. Ein Teilnehmer gab an, dass lediglich 80% der Geräte verschlüsselt werden, obwohl die Sollquote bei 100% liege.

Im Smartphonebereich scheint die Verschlüsselung auf den ersten Blick nicht ganz so verbreitet: 68% aller Firmen aktivieren Verschlüsselungsfunktionen vor der Ausgabe von Smartphones an ihre Mitarbeiter. Hierbei ist jedoch zu berücksichtigen, dass viele Firmen keine sensitiven Daten auf Smartphones dulden. Lediglich 3% der Firmen die angaben, sensitive Daten auf Smartphones zu speichern, verschlüsseln diese nicht. Zu denken gibt aber, dass wie im privaten Bereich, die Passwortstärke auf Smartphones oft zu gering ist. Analog zu Tabelle 3 vergeben nur 19% der Unternehmen sichere Passwörter (d.h. mindestens 8 alphanumerische Zeichen).

3.2.3 Entscheidungsgründe und Einschätzungen

Von der Möglichkeit ihre Entscheidung für oder gegen den Einsatz einer Verschlüsselungsmethode zu begründen, machten 24 Teilnehmer (73%) Gebrauch. Für die Verwendung wurden hauptsächlich zwei Argumente genannt: Schutz der eigenen Daten, sowie Vorgaben durch Sicherheitsrichtlinien und gesetzliche Vorschriften (vgl. Tabelle 7).

Datensicherheit	12
Sicherheitsrichtlinien/Vorschriften	6
Standard	2
Active Directory	1
Imageschutz	1
Schutz vor Schadensersatzklagen	1

Tabelle 7: Gründe für Verschlüsselung i.A.

Administrierbarkeit/Flexibilität	14
Kosten	8
bessere Wiederherstellbarkeit	2
SED Technik unausgereift	1
mangelnde Erfahrung mit SEDs	1
Software für Nutzer verständlicher	1

Tabelle 8: Gründe für SW-basiertes FDE.

Für eine softwarebasierte Verschlüsselung sprechen aus Firmensicht vor allem zwei Dinge (vgl. Tabelle 8): die Flexibilität und bessere Administrierbarkeit, sowie finanzielle Erwägungen. Vereinzelt tauchen darüberhinaus Argumente auf, die für ein noch nicht vorhandenes Vertrauen in die Technologie der SEDs sprechen.

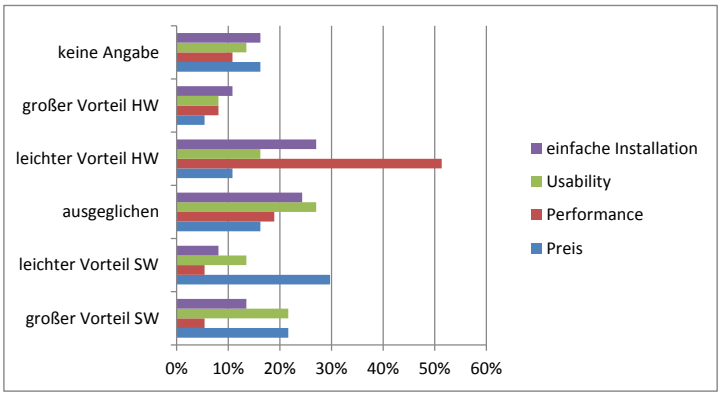


Abbildung 8: Vergleich software- und hardwarebasierter Verschlüsselung.

Ebenso wie die Privatanwender, wurden auch die geschäftlichen Teilnehmer nach ihren Einschätzungen bezüglich der Sicherheit und den Vor- und Nachteilen von hardware- bzw. softwarebasierter Verschlüsselung gefragt (vgl. Abbildung 8). Die Usability und die Installation von SEDs gegenüber softwarebasierterer Verschlüsselung wird von den Befragten dabei als ausgeglichen angesehen, wohingegen der Performancevorteil deutlich bei hardwarebasierter Verschlüsselung gesehen wird und der Preisvorteil bei softwarebasierter Verschlüsselung.

Ferner wurde gefragt, nach welchen Kriterien die Betriebe neue Hardware bzw. Software erwerben. Die wichtigen und sehr wichtigen Kriterien für den Einkauf sind demnach: Usability (62%), Performance (51%), einfache Inbetriebnahme (51%), Preis (46%) und das

Image des Herstellers (41%) bzw. ob bereits eine Kooperation mit dem Hersteller besteht (35%). Der Kostenfaktor spielt für viele Firmen also nicht die wichtigste Rolle, sondern vielmehr die Usability und Performance. Knapp 2/3 der Unternehmen sind dazu bereit Geschäftsbeziehungen mit neuen Partnern einzugehen.

Abschließend wurde untersucht, ob die in Kapitel 2.2 beschriebenen Angriffe bekannt sind. Dabei ist festzustellen, dass die Angriffe durchgehend bekannter sind als im Bereich der privaten Anwender. Dies ist darauf zurückzuführen, dass die Fragebögen durch Administratoren und IT-Verantwortliche der Firmen ausgefüllt wurden; das Wissen der Mitarbeiter über solche Angriffe wird dadurch nicht wiedergespiegelt. Cold-Boot Angriffe waren demnach 65% der Firmen bekannt, FireWire bzw. DMA Angriffe 51%, Hot-Plug Angriffe 43% und Evil-Maid Angriffe 30%.

4 Fazit und Ausblick

Im geschäftlichen Umfeld ist die Verschlüsselung von Laptops weiter verbreitet als im privaten Umfeld. 85% der untersuchten Firmen gaben an Verschlüsselungstechnologien zum Schutz ihrer Daten einzusetzen, wobei nahezu alle Unternehmen auf Softwarelösungen setzen und nur vereinzelt begonnen wurde zusätzlich SEDs einzusetzen. Bevorzugt werden zur Verschlüsselung kostenpflichtige Lösungen bekannter Hersteller, wobei Microsoft BitLocker mit einem Marktanteil von 50% hervorzuheben ist. Die Smartphoneverschlüsselung erscheint gegenüber der Laptopverschlüsselung mit 68% etwas geringer, wobei aber einige Firmen keine sensitiven Daten auf Smartphones speichern.

Die Zahlen unserer Umfrage für deutsche Unternehmen decken sich in etwa mit den eingangs erwähnten amerikanischen Studien. Im Detail weichen diese zwar voneinander ab, aber gemeinsame Tendenzen sind deutlich erkennbar. Einzigst die auffallend niedrige Verlustrate von 0,7% für Laptops in deutschen Firmen, wie sie unsere Untersuchung ergeben hat, sticht hervor.

Zudem hat unsere Studie gezeigt, dass die private Verwendung von FDE mit 25% weit weniger verbreitet ist. Meist wurde der Nichteinsatz von Verschlüsselung mit einem Mangel an Notwendigkeit begründet, aber es zeigte sich auch, dass viele Teilnehmer zu wenig über Verschlüsselung wissen und den Aufwand einer Installation scheuen. Da davon auszugehen ist, dass bei der Stichprobengruppe ein gegenüber der Gesamtbevölkerung erhöhtes technisches Wissen und Interesse vorzufinden ist, nehmen wir an, dass die deutsche Allgemeinheit ihre Daten seltener verschlüsselt. Bei privaten Smartphones ist die Verschlüsselungsrate mit 42% wesentlich höher als bei Laptops. Dies liegt mutmaßlich an der leichten Verfügbarkeit von Verschlüsselungstechnologien auf modernen Smartphones, welche in diesem Punkt als Vorbild für zukünftige Laptops dienen können.

Auch Privatanwender nutzen, sofern sie denn verschlüsseln, meist ein softwarebasiertes Produkt, insbesondere die Open-Source Lösung TrueCrypt. SEDs spielen heute sowohl im geschäftlichen als auch im privaten Umfeld eher eine untergeordnete Rolle. Wie unsere Untersuchungen ergeben haben, sind die Gründe hierfür nicht ausschließlich finanzielle Aspekte, sondern es fehlt mitunter an Vertrauen in diese neue Technologie.

Literatur

[DBK05] Maximillian Dornseif, Michael Becher und Christian N. Klein. FireWire - All your memory are belong to us. In *Annual CanSecWest Applied Security Conference*, Vancouver, British Columbia, Canada, 2005. Laboratory for Dependable Distributed Systems, RWTH Aachen.

[GM13] Johannes Götzfried und Tilo Müller. Evil Maid goes after Android: How to subvert Android smartphones with keylogging. Bericht, Universität Erlangen-Nürnberg, 2013.

[HSH+08] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum und Edward W. Felten. Lest We Remember: Cold Boot Attacks on Encryptions Keys. In *17th USENIX Security Symposium*, Seiten 45–60, San Jose, CA, August 2008. Princeton University, USENIX.

[MLF13] Tilo Müller, Tobias Latzo und Felix Freiling. Self-Encrypting Disks pose Self-Decrypting Risks: How to break Hardware-based Full Disk Encryption. Bericht, Universität Erlangen-Nürnberg, Dezember 2013.

[MS13] Tilo Müller und Michael Spreitzenbarth. FROST: Forensic Recovery of Scrambled Telephones. In *11th International Conference on Applied Cryptography and Network Security (ACNS '13)*, Banff, Alberta, Canada, Juni 2013. Universität Erlangen-Nürnberg, Springer.

[Pon10] Ponemon. *2010 Annual Study: U.S. Enterprise Encryption Trends*. November 2010.

[Rut09] Joanna Rutkowska. Evil Maid goes after TrueCrypt. http://theinvisiblethings.blogspot.com/2009/10/evil-maid-goes-after-truecrypt.html, Oktober 2009. The Invisible Things Lab.

[SEC12] SECUDE. *US Full Disk Encryption 2011 Survey*. Research SECUDE AG, Marz 2012.

Angriffe auf eine Spreizspektrummethode für Audio-Steganographie

Andreas Westfeld
Hochschule für Technik und Wirtschaft Dresden
01069 Dresden, Friedrich-List-Platz 1

Abstract: Steganographie ist eine Technik zur vertraulichen Kommunikation. Ihre Sicherheit definiert sich über das Unvermögen des Angreifers, die Existenz einer vertraulichen Kommunikation nachzuweisen.

Ziel dieses Beitrags ist es, einige Analysemuster für Schwachstellen am Beispiel einer veröffentlichten Spreizspektrummethode für Steganographie in Audiomedien (Nutzinger und Wurzer, ARES 2011) vorzustellen. Einige Schwächen werden gefunden und beseitigt.

1 Einführung

Steganographie ist die Kunst und Wissenschaft der verdeckten Kommunikation. Ihr Zweck ist die Übertragung von Informationen, die in einem Trägermedium sicher versteckt sind. Sichere *Wasserzeichen*verfahren betten zwar kurze Nachrichten robust ein. Denn diese sollen bis zu einem gewissen Grade gegen verändernde Angreifer, etwa Nutzer, die das Wasserzeichen entfernen wollen, geschützt werden. In der Regel hinterlassen Wasserzeichenverfahren jedoch leicht nachweisbare Spuren. Dagegen lässt sich zwar die Existenz sicherer *steganographischer* Nachrichten nicht nachweisen, üblicherweise sind diese jedoch selten robust eingebettet. Denn der Empfänger der steganographischen Nachricht hat selbst grundsätzlich kein Interesse, die Nachricht vor dem Empfang zu stören. Für hinreichenden Schutz auf dem Übertragungsweg kann die Netz-Sicherungsschicht sorgen – unabhängig davon, ob eine Nachricht eingebettet wurde und mit welcher Robustheit.

Bei Internet-Telefonie ist eine möglichst geringe Verzögerungszeit wichtiger als eine vollständige Fehlerkorrektur. Zugunsten einer niedrigen Latenz wird unbestätigt übertragen. Verlorene, verzögerte oder fehlerhaft übertragene Pakete werden nicht erneut gesendet und verlangsamen daher die Laufzeit nicht. Wegen der Redundanz des gesprochenen Wortes und durch die fehlermindernde Wirkung der Codierung ist ein Telefonat dennoch ohne nennenswerte Beeinträchtigung möglich. Auch bei einigen Funkbetriebsarten (z. B. analoger Sprechfunk, Schmalbandfernsehen) wird auf Fehlerkorrektur verzichtet, weil sich die entstehenden Fehler nur geringfügig auf das Trägermedium auswirken und toleriert werden können.

Überwiegend können wir bei digitaler steganographischer Kommunikation davon ausgehen, dass Nachrichten ungestört empfangen werden. So erreichen z. B. digitalisierte Bilder, die als E-Mail-Anhang versendet werden, praktisch immer fehlerfrei den Empfänger. Wenn jedes Bit des Trägermediums den Empfänger ungestört erreicht, dann lässt sich auch eine eventuell eingebettete steganographische Nachricht problemlos extrahieren.

Ohne Fehlerkorrektur (z. B. Internet-Telefonie, analoge Funkbetriebsarten) sind Störungen nur dort erträglich, wo sie die geringsten Auswirkungen auf das Trägermedium haben, also an den irrelevanten, für unsere Sinne schlecht wahrnehmbaren Stellen eines Trägermediums. Typische steganographische Algorithmen betten jedoch mit Vorliebe in solche Stellen ein. Die eingebettete Nachricht wäre also in der Gegenwart von Störungen dort am meisten gefährdet. Deshalb müssen robuste Einbettungsfunktionen die Auswahl der änderbaren Stellen in Bezug auf das Verhältnis zwischen unauffälliger Änderbarkeit und Fehlergefahr optimieren sowie durch Hinzufügen von Redundanz möglichen Störungen vorbeugen. Beide Maßnahmen bewirken in ihrer Konsequenz geringere Kapazität und erhöhte Entdeckungsgefahr.

Dieser Beitrag beschäftigt sich mit einem steganographischen Verfahren, das dennoch robust ist, nicht gegen absichtliche Störungen eines Angreifers, aber gegen unbeabsichtigte, zufällige Kanalfehler, wie sie im Zusammenhang mit Telefonie auftreten.

Diese im Vergleich zu digitalen Wasserzeichen beschränkte Robustheit ermöglicht höhere Kapazität, z. B. verglichen mit dem Verfahren von Tachibana und Kollegen [TSNK01] eine viermal größere geheime Nachricht in einem Trägermedium, das nur ein Fünfzigstel der Bandbreite benötigt. Wie es um die Sicherheit bestellt ist, soll in diesem Papier näher untersucht werden.

Beispiele für robuste Steganographie sind eher selten und verwenden in den meisten Fällen digitale Bilder als Trägermedium. Marvel und Kollegen entwickelten ein robustes steganographisches Verfahren für Bilder [MBR99] auf der Basis von Spreizspektrum-Modulation [PSM82]. Damit lassen sich Informationen unterhalb des Störpegels übertragen. Weitere Beispiele sind Verfahren für Slow-Scan-Television-Signale (SSTV) [Wes06] oder in Audiosignalen.

Dieser Beitrag untersucht ein Spreizspektrumverfahren für Steganographie in Audiomedien, das von Nutzinger und Kollegen 2010 eingeführt [NFM10] und von Nutzinger und Wurzer 2011 implementiert wurde [NW11]. Dieses Verfahren überlebte verschiedene Robustheitstests seiner Entwickler wie additives Rauschen, variable Zeitverzögerung, variable Frequenzverschiebung, GSM-Kodierung, akustische Übertragung, Neuabtastung und Schnitt. Es verursachte auch keine nennenswerte Veränderung der wahrnehmbaren Störungen bei Hörtests im Vergleich zum Originalsignal [NW11].

Ziel dieses Beitrags ist, wie der Titel nahelegt, nicht die Beschreibung oder der Nachweis der Sicherheit eines neuen steganographischen Verfahrens, sondern die Demonstration häufig auftretender Schwächen am Beispiel einer bereits in der Literatur evaluierten Implementierung, wenngleich die vorgeschlagene Mängelbeseitigung unter bestimmten

Voraussetzungen zu einem brauchbaren Ergebnis führen kann. Einige dieser Voraussetzungen können wir als Angreifer bestimmen. Dennoch sagt das wenig über die Risiken bei einem ernsthaften Einsatz des Verfahrens. Es ist gut möglich, dass sich einige der hier beschriebenen Angriffe erfolgreich auf andere steganographische (Audio-)Methoden übertragen lassen. Dennoch waren die Entwickler sich dieser Angriffe bei ihrer eigenen, nicht minder aufwändigen Evaluation nicht bewusst. Zwar kann der Steganograph manche Schwäche ausräumen, indem er eine geeignete Trägermedien-Quelle wählt. Dennoch ist beim vorliegenden Sicherheitsniveau kein universeller Detektor nötig, der sämtliche Quellen unterscheiden kann. Es ist immer ratsam, die Ursache der Schwächen zu ermitteln und die dafür verantwortlichen Teile zu überarbeiten.

Im folgenden Abschnitt werfen wir zunächst einen Blick auf das Spreizspektrumverfahren und seine Motivation, stellen anschließend die gefundenen Schwächen vor, die in Abschnitt 4 beseitigt werden. Abschnitt 5 beschließt den Beitrag.

2 Spreizspektrumverfahren

Der steganographische Algorithmus nutzt das Audiosignal eines Telefongesprächs als Trägermedium. Es ist zunächst nicht wichtig, ob es ein VoIP-Telefonat ist oder eine mobile Konversation über GSM oder UMTS, und die Anwendung ist auch nicht auf solche Audiosignale beschränkt. Die Nachricht wird in die dekodierten Audiodaten eingebettet. Die ursprüngliche Idee war, direkt in die analoge Information einzugreifen, aus praktischen Gründen also so nah wie möglich an der Quelle anzusetzen. Damit verband man wohl die Hoffnung, dass Schwächen der Steganographie, die überwiegend durch unvollkommene digitale Einbettungsfunktionen verursacht werden, z. B. das Überschreiben niederwertigster Bits, vermieden werden.

Beim Spreizspektrumverfahren wird die einzubettende Information in ein möglichst breitbandiges, dafür sehr schwaches pseudozufälliges Signal (Pseudorauschen, pseudo noise, PN-Folge), über einen längeren Zeitraum gespreizt. Nun ist die Bandbreite in dem für Telefonate typischen Übertragungskanal stark beschränkt. Das „breitbandige“ Rauschen wird, da ein Informationsverlust nicht gewollt ist, entsprechend schmalbandig ausfallen und vor allem über die Zeit gespreizt. Die Beschreibung bleibt an dieser Stelle aus dramaturgischen Gründen offen, weil am Anfang der Untersuchung kaum Informationen – zumindest nicht die konkrete Implementierung [NW11] – zur Verfügung standen.

3 Angriffe

Vor der Suche nach hinterlassenen Einbettungsspuren ist es sinnvoll, zunächst möglichst viele Informationen über das Verfahren zusammenzutragen. Der Verfasser musste je-

doch (unfreiwillig) den Angreifer in zwei unterschiedlichen Situationen spielen. Es war zunächst unklar, ob der Auftraggeber des Einbettungsprogramms der Untersuchung des Programms zustimmen würde. So war in der ersten Phase zunächst weder der C++-Quelltext noch die ausführbare Datei verfügbar. Lediglich eine kleine Anzahl aufgezeichneter Telefonate (WAV-Dateien) stand in verschiedenen Versionen zur Verfügung:

- ohne eingebettete Nachricht (Original) und
- mit eingebetteter Nachricht unter Verwendung verschiedener Konfigurationen (bezeichnet mit „amp“, „bpsk“ und „phase“) unter welchen die beste herausgefunden werden sollte.

Es war bekannt, dass die Nachrichten mit einer Art Spreizspektrumverfahren eingebettet wurden.

3.1 Zwillingsspitzen?

Soweit die genaue Implementierung unbekannt war, wurde ein einfacher DSSS-Algorithmus (direct sequence spread spectrum) unterstellt. Der Verfasser hoffte, dass wenigstens eine der drei Konfigurationen („amp“, „bpsk“ und „phase“) nahe genug an dieser sicherlich vereinfachenden Annahme liegt. Abbildung 1 zeigt ein konkretes Beispiel für

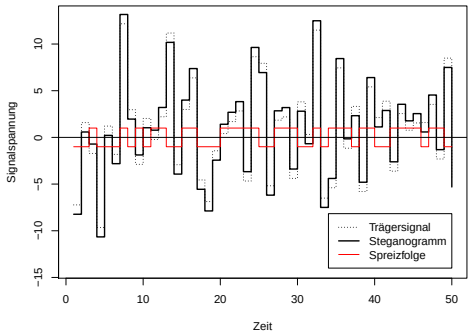


Abbildung 1: Eine pseudozufällige Spreizfolge wird mit dem Trägersignal (gepunktete Linie) überlagert und ergibt so das Steganogramm (fette Linie)

die unterstellte, einfache Spreizsprektrum-Einbettung mit einem normalverteilten Trägersignal. Eine pseudozufällige Spreizfolge, die ausschließlich aus den Werten -1 und 1 besteht, wird dabei verwendet, um ein einzubettendes Symbol (z. B. ein Bit) über einen längeren Zeitabschnitt zu spreizen. Obwohl diese Spreizfolge mit dem Trägersignal überlagert wird (bezogen auf die eingebettete Nachricht ein Störsignal), lässt sich das Symbol anschließend als Skalarprodukt der Spreizfolge mit dem Steganogramm extrahieren.

Passend zur einfachen Spreizspektrum-Einbettung gibt es einen einfachen Angriff in der Literatur, nämlich einen Histogramm-Angriff namens *Twin Peaks* [Mae98]. Wenn im Histogramm des Trägermediums eine Spitze vorhanden ist, könnte sich im Histogramm des Steganogramms eine verräterische Zwillingsspitze zeigen, woher der Name des Angriffs rührt. Wenn das Trägermedium eine Normalverteilung aufweist, dann ist das Steganogramm die Summe von zwei gegeneinander verschobenen Normalverteilungen, wobei der Abstand zwischen beiden durch die Spreizfolge bestimmt wird ($1 - (-1) = 2$). Wenn die Streuung des Signals groß ist (vgl. Abb. 2, links), z. B. an den lauteren Stellen des Tele-

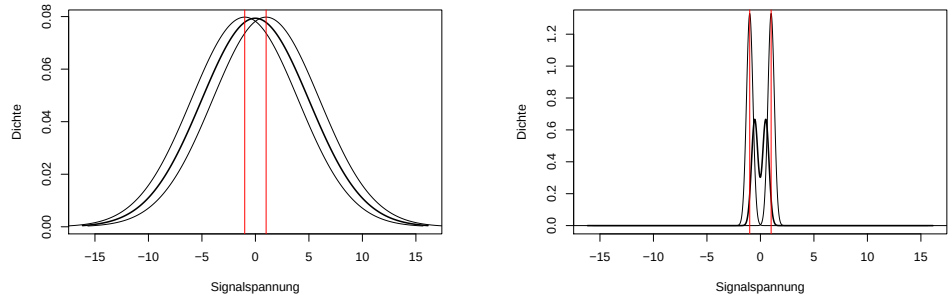


Abbildung 2: Wenn das Trägersignal lautstark genug ist, scheint die zusammengesetzte Verteilung (fette Linie) ebenfalls normalverteilt zu sein (links), leise Passagen des Steganogramms könnten jedoch eine Zwillingsspitze zeigen (rechts).

fonats, ist die sich ergebende Verteilung kaum von der ursprünglichen Normalverteilung zu unterscheiden. In leiseren Passagen könnte sich jedoch eine bimodale Verteilung ergeben (vgl. Abb. 2, rechts) oder zumindest eine nennenswerte Veränderung der Verteilung, an der die steganographische Einbettung erkennbar ist. Natürlich können wir erwarten, dass eine zeitgemäße Einbettungsmethode mit einer automatischen Pegelregelung für die Spreizsequenz aufwartet, die deren Amplitude in Abhängigkeit vom Trägersignal dynamisch verringert. Andererseits soll die Einbettungsmethode auch für Telefonate geeignet sein und muss sich zugunsten der Echtzeitfähigkeit einschränken. Die Regelung kann z. B. keinen allzu langen Abschnitt des Signals betrachten, ohne störende Verzögerungen bei der Übermittlung des Sprachsignal zu verursachen oder selbst verzögert zu reagieren.

Überraschenderweise ließen sich bei dieser ersten Untersuchung selbst kurze Ausschnitte der Telefonate (ca. 2–10 Sekunden) perfekt auseinanderhalten. Der Detektor arbeitet in zwei Schritten. Im ersten wurden leise Stellen des Signals ausgewählt, denn an den lauten erwarten wir ja gerade keine Zwillingsspitzen. Später stellte sich heraus, dass dieser Schritt verzichtbar ist. Im zweiten wurde das Histogramm gebildet, das eine „Solospitze“ für den Signalwert 0 zeigte, jedoch nur im Steganogramm. Das Trägersignal enthielt ungefähr die gleiche Anzahl Nullen und Einsen (ggf. bis zu 30 % mehr), in Steganogrammen gab es doppelt so viele. Der Angriff funktionierte in allen Konfigurationen („amp“, „bpsk“ und „phase“) mit dem gleichen Schwellwert: Wenn es mehr als 1,5 Mal so viele Nullen wie Einsen gab, dann wurde das Signal als „steganographisch verändert“ bewertet.

Warum das so war, blieb verborgen, bis letztlich der Quelltext des Verfahrens verfügbar war (vgl. Abschnitt 4.1).

3.2 „Stufen“

Dank der verfügbaren Testdaten jeweils mit und ohne eingebettete Nachricht, ist hier eine Analyse durch synchrone Gegenüberstellung der abgetasteten Werte c_i bzw. s_i des Trägermediums bzw. des Steganogramms möglich. Je näher die abgetasteten Werte an der Diagonale $s_i = c_i$ liegen, desto geringfügiger war die steganographische Veränderung des Signals. Abbildung 3 stellt Abtastwerte von Trägermedium und zugehörigem Stega-

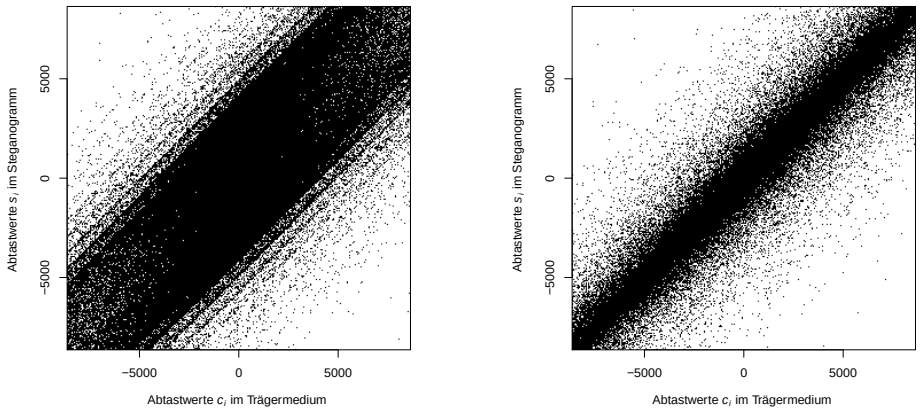


Abbildung 3: Die synchrone Gegenüberstellung der Abtastwerte im Trägermedium und im Steganogramm zeigt Steuerstufen (links), jedoch nicht mehr nach Korrektur des Algorithmus (rechts)

nogramm gegenüber, wobei eine Überlagerung diagonaler Streifen sichtbar wird. Offensichtlich gibt es eine Regelung, die die Einbettungsintensität in diskreten Schritten steuert. Unter realistischen Bedingungen ist dieser direkte Vergleich jedoch kaum möglich, da das Trägersignal nicht mit übertragen wird. Das fehlende Trägersignal kann dann nur aus dem steganographischen geschätzt werden. Dieser Prozess wird „Entrauschen“ genannt.

3.3 Saturn gesichtet

Die Abtastwerte des Trägermediums könnten aus anderen, jedoch zeitlich naheliegenden steganographisch veränderten Werten geschätzt werden:

$$c_i \sim s_{i-2} + s_{i-1} + s_{i+1} + s_{i+2}. \tag{1}$$

Ein ähnlicher Ansatz wurde während des BOWS-2-Wettbewerbs erfolgreich genutzt, um die Größenordnung von Wavelet-Koeffizienten ohne Wasserzeichen aus der Umgebung zu schätzen [BW09]. Dies war möglich, da der Teil des Wasserzeichens in s_i *unabhängig* vom Wasserzeichen in den Werten der Umgebung war. Leider können wir bezüglich der Unabhängigkeit im Falle des angegriffenen Audiosignals nicht sicher sein oder diese Eigenschaft gar erwarten, da eine Chipzeit – das ist die zeitliche Länge der Spreizfolge eines Symbols – in der Regel länger ist als die Abtastperiode. Dennoch können wir einfach mal den nächsten unveränderten Abtastwert aus dessen steganographischem Vorgänger schätzen

$$\hat{c}_{i+1} = s_i, \tag{2}$$

was uns zu einer eindrucksvollen, „astronomischen“ Konstellation in Abb. 4 (links) führt.

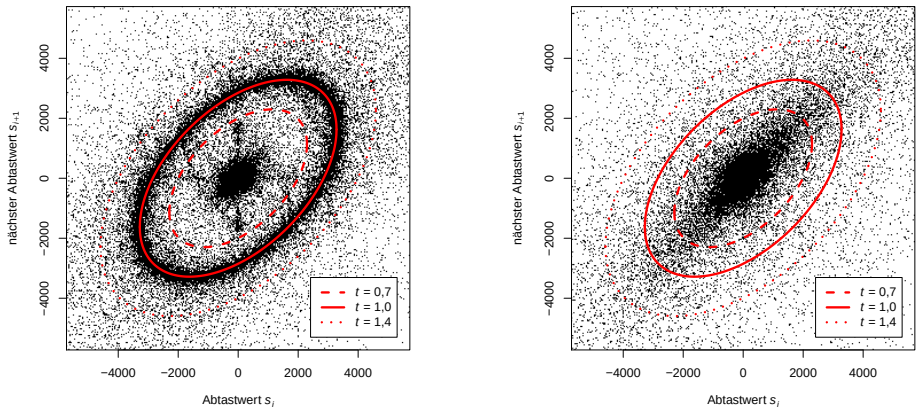


Abbildung 4: Die Gegenüberstellung aufeinanderfolgender Abtastwerte im Steganogramm (nächster Abtastwert als eine Schätzung des Trägersignals) zeigt eine Lissajous-Figur (links), nicht jedoch nach Reparatur des Algorithmus (rechts)

Obwohl die Ähnlichkeit mit einem Gestirn unseres Sonnensystems nicht abzustreiten ist, handelt es sich – technisch gesehen – um eine Lissajous-Figur, die durch folgende Ellipse

abgeschätzt werden kann:

$$\begin{aligned} t^2 &= \left(\frac{y-mx}{b}\right)^2 - \frac{x^2}{a^2}, \quad \text{mit} \\ a &= 3280, \\ b &= 2850 \quad \text{und} \\ m &= 0{,}496, \end{aligned} \tag{3}$$

wobei t ein Schwellwert ist, der bei geeigneter Wahl eine Unterscheidung zwischen Ring und Zentrum ermöglicht. Lissajous-Kurven erscheinen auf einem Oszilloskop im X–Y-Betrieb, also bei abgeschalteter Zeitablenkung, wenn an beide Eingänge je ein sinusförmiges Signal angelegt wird. In unserem Fall stammen beide Signale aus der selben Quelle. Sie haben also die gleiche Frequenz, sind aber untereinander zeitverzögert. Die Zeitverzögerung oder Phasenverschiebung wird im vorliegenden Fall durch die Abtastperiode festgelegt. Ein möglicher Detektor könnte den fragwürdigen Anteil von Abtastwerten ermitteln, der zwischen der gestrichelten Ellipse ($t = 0{,}7$) und der gepunkteten ($t = 1{,}4$) auftritt, und ihn mit der Gesamtzahl aller Werte vergleichen.

Die Stärke des Effekts ist sicherlich überraschend. Im Telefonat ist diese Einbettungsintensität auch deutlich als Pfeifen hörbar. Das Signal wurde ursprünglich zum Test der steganographischen Übertragungseigenschaften in GSM-Kanälen erzeugt und fand so seinen Weg zu den Testdaten. Der Effekt ist jedoch auch beim Betrieb mit realistischeren Parametern als kleiner Ring ($a \approx b \approx 30$: ;-) in einigen leisen Passagen vorhanden (jedoch sonst von dem in Abb. 4 sichtbaren dunklen Zentrum überdeckt). Wahrscheinlich wären wir ohne dieses Versehen gar nicht darauf aufmerksam geworden.

4 Reparatur

4.1 Solospitze

Das Trägersignal wird aus einer Audioquelle gelesen, gewöhnlich ein Telefon, eine Soundkarte oder eine WAV-Datei (z. B. 8000 PCM 16-Bit-Abtastwerte pro Sekunde, Mono). Die Werte sind vorzeichenbehaftete ganze Zahlen. Der Einbettungsalgorithmus unterstützt verschiedene Abtastraten und Genauigkeiten. Er wandelt daher anfänglich die Rohdaten in eine Folge normalisierter `double`-Gleitkommawerte im Bereich $-1 \ldots 1$ um.

Implementiert ist das durch Typumwandlung von `double` nach `int` (Ganzzahl), gefolgt von einer Skalierung in das Interval $[-1, 1]$. Am Ende werden die `double`-Werte wieder in den ursprünglichen Bereich hochskaliert (und, falls nötig, beschnitten) sowie nach `int` zurückgewandelt. Solange die Werte dazwischen nicht durch den Einbettungsschritt

verändert werden, ist die Rückwandlung von `double` eine $1:1$-Abbildung, da neben dem ganzzahligen Teil kein Bruchteil bleibt.

Wenn jedoch etwas eingebettet wird, dann sind die entstehenden `double`-Werte in der Regel nicht ganzzahlig. Hier rächt sich der naheliegende, aber sorglose Einsatz der Typumwandlung. Der Typumwandlungs-Operator (`y = (int)x;`) erwartet einen numerischen Parameter $x \in \mathbb{R}$ und gibt eine Ganzzahl $y \in \mathbb{Z}$ zurück, die durch Abschneiden des Bruchteils gebildet wird (vgl. Abb. 5). Mögliche Reparatur:

```
y = (int)(x + ((x < 0) ? -0.5 : 0.5));
```

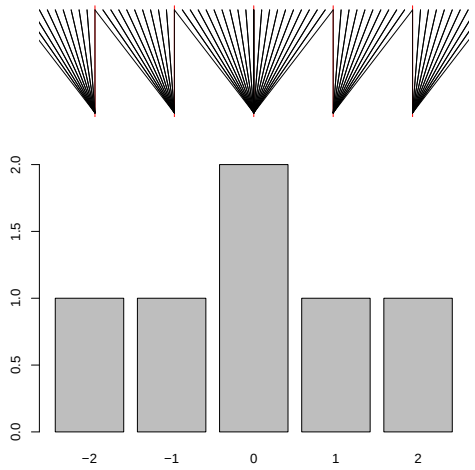


Abbildung 5: Im Korb für die 0 sammeln sich wegen des Abschneidens des Bruchteils die Werte aus einem doppelt so breiten Interval, wodurch die Spitze im Histogramm entsteht

Nachdem das eine Problem der auffälligen Solospitze behoben war, fiel bei einigen Medien noch ein weiteres im Sättigungsbereich auf. Es gibt eine gerade Anzahl von (vorzeichenbehafteten) 16-Bit-Ganzzahlen. Eine ist neutral (0), 32767 sind positiv und 32768 haben einen negativen Wert. Bei der Vorzeichenumkehr von -32768 (`0x8000`) kommt es zu einem Vorzeichenüberlauf: Es entsteht derselbe negative Wert. Die implementierte Normalisierung teilte alle Werte durch 32767 und beschnitt bei -32768 auf $-1{,}0$. Wenn das Signal nun hinreichend gesättigt ist, entstehen im Histogramm jeweils Spitzen für die gesättigten Werte $(-32768, 32767)$. Die Umwandlung in den internen Gleitkommabereich und zurück verschiebt dann die Spitze von -32768 nach -32767. Eine Spitze bei -32767 ist zwar ein zuverlässiger Indikator für steganographische Einbettung, tritt jedoch nur bei gesättigtem Ausgangsmaterial auf. Der Mangel lässt sich offensichtlich reparieren, indem der Divisor auf 32768 geändert wird und positive Werte bei nach der Rückumwandlung auf 32767 begrenzt werden.

5 Zusammenfassung

Wir haben mehrere Schwächen in einem Spreizspektrumverfahren für Audiosteganographie gefunden. Einige davon sind nicht auf die Einbettung selbst zurückzuführen, sondern die Umwandlung zwischen einem externen Format und einem internen Arbeitsformat. Es scheint also besonders wichtig zu sein, Umwandlungs- und Normalisierungsfunktionen auf Homogenität um besondere Werte wie 0 und ihre Eigenschaften bei Sättigung zu überprüfen.

Aber auch das Einbettungsverfahren selbst zeigte Schwächen. Es ist offenbar ratsam, die Differenz zwischen dem unveränderten und dem erzeugten Signal beim Entwurf genau zu analysieren. Zwar hat ein typischer Angreifer keinen Zugriff auf diese Differenz, auffällige Eigenschaften können dennoch über die abschirmende Wirkung der Unbestimmtheit aufgrund des zum Vergleich fehlenden Trägermediums hinausleuchten. Auch ist es ratsam, pathologische Signale wie rhythmische Audio-Impulse z. B. beim Test von Steuerungen einzusetzen.

Ferner ist es unerlässlich, Korrelationen innerhalb der Elemente des Trägermediums zu berücksichtigen. Solche Korrelationen treten natürlich auch im Steganogramm auf. Sie können häufig ausgenutzt werden, um das Signal zu „entrauschen“ oder entscheidende Statistiken zu „kalibrieren“ [FGH03, KF09], selbst in Audioströmen.

Literatur

[BW09] Patric Bas und Andreas Westfeld. Two Key Estimation Techniques for the Broken Arrows Watermarking Scheme. In *Proc. of ACM Multimedia and Security Workshop*, Seiten 1–8, Princeton, NJ, USA, September, 7–8 2009.

[FGH03] Jessica Fridrich, Miroslav Goljan und Dorin Hogea. Steganalysis of JPEG Images: Breaking the F5 Algorithm. In Fabien A. P. Petitcolas, Hrsg., *Information Hiding (5th International Workshop)*, Jgg. 2578 of *LNCS*, Seiten 310–323, Berlin Heidelberg, 2003. Springer-Verlag.

[KF09] Jan Kodovský und Jessica Fridrich. Calibration Revisited. In *Proc. of ACM Multimedia and Security Workshop*, Seiten 63–73, Princeton, NJ, USA, September, 7–8 2009.

[Mae98] Maurice Maes. Twin Peaks: The Histogram Attack to Fixed Depth Image Watermarks. In David Aucsmith, Hrsg., *Information Hiding (2nd International Workshop)*, Jgg. 1525 of *LNCS*, Seiten 290–305, Berlin Heidelberg, 1998. Springer-Verlag.

[MBR99] Lisa M. Marvel, Charles G. Boncelet und Charles T. Retter. Spread Spectrum Image Steganography. *IEEE Transactions on Image Processing*, 8:1075–1083, 1999.

[NFM10] Marcus Nutzinger, Christian Fabian und Marion Marschalek. Secure Hybrid Spread Spectrum System for Stegnanography in Auditive Media. *6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, Seiten 78–81, October 2010.

[NW11] Marcus Nutzinger und Jürgen Wurzer. A Novel Phase Coding Technique for Steganography in Auditive Media. *6th International Conference on Availability, Reliability and Security (ARES)*, Seiten 91–98, August 2011.

[PSM82] Raymond L. Pickholtz, Donald L. Schilling und Laurence B. Milstein. Theory of Spread-Spectrum Communications—A Tutorial. *IEEE Transactions on Communications*, 30:855–884, 1982.

[TSNK01] Ryuki Tachibana, Shuichi Shimizu, Taiga Nakamura und Seiji Kobayashi. An Audio Watermarking Method Robust Against Time- and Frequency-Fluctuation. In Edward J. Delp III und Ping Wah Wong, Hrsg., *Security, Steganography and Watermarking of Multimedia Contents III (Proc. of SPIE)*, Seiten 104–115, San Jose, CA, January 2001.

[Wes06] Andreas Westfeld. Steganographie für den Amateurfunk. In Jana Dittmann, Hrsg., *Sicherheit 2006, Sicherheit – Schutz und Zuverlässigkeit, Beiträge der 3. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e. V. (GI), 20.–22. Februar 2006 in Magdeburg*, Jgg. P-77 of *LNI*, Seiten 119–130, Bonn, 2006.

Lässt sich der Schreibstil verfälschen um die eigene Anonymität in Textdokumenten zu schützen?

Oren Halvani, Martin Steinebach, Svenja Neitzel*

Fraunhofer SIT, Darmstadt, {Halvani, Steinebach}@SIT.Fraunhofer.de
*TU Darmstadt, Svenja.Neitzel@Freenet.de

Abstract: Die Zahl textueller Daten wächst heutzutage zunehmend, insbesondere aufgrund nutzergenerierter Inhalte im Internet. Zu diesen zählen unter anderem Blogs, Forenbeiträge oder Kommentare, die über unzählige Plattformen verbreitet werden. Wünscht ein Autor hier anonym zu kommunizieren, nutzt er ein oder mehrere Pseudonyme. Schreibstile dagegen verbleiben ungeschützt in den Texten und können mit Hilfe sogenannter Autorschafts-Attributionssysteme bekannten Autoren zugeordnet werden. Aktuelle Systeme erzielen dabei je nach Szenario (Anzahl der Autoren, Qualität der Daten, etc.) gute bis sehr gute Ergebnisse. Wenn die Möglichkeit der Anonymität angestrebt wird, ist folglich eine wichtige Frage, ob und wie Schreibstile in Texten verfälscht werden können, um solche Systeme zu täuschen. In diesem Papier werden zunächst Systeme und deren Komponenten erläutert, mit deren Hilfe Texte hinsichtlich der darin enthaltenen Schreibstile de-anonymisiert werden können. Anschließend wird ein Überblick über manuelle und semi-automatische Gegenmaßnahmen gegeben. Weiterhin werden Möglichkeiten genannt, um eine vollautomatische Anonymisierung der Schreibstile zu realisieren.

1 Einführung

Dank des Internets ist die Zahl existierender Textdaten massiv angestiegen, nicht zuletzt aufgrund nutzergenerierter Inhalte. Dazu zählen vor allem Blogs, Forenbeiträge und Kommentare zu mehr oder weniger brisanten Themen. Autoren, die sich durch die Verwendung von Pseudonymen hinsichtlich ihrer Anonymität sicher fühlen, üben so teilweise sehr offen Kritik. So finden sich beispielsweise auf dem Portal *Jameda.de* ca. $3 \cdot 10^6$ Kommentare von Patienten über deren behandelnde Ärzte. Zum einen können so im Schutz der Anonymität berechtigte Warnungen vor Ärzten mit (zumindest subjektiv empfundenen) schlechten Behandlungsmethoden ausgesprochen werden. Zum anderen besteht die Gefahr, dass die Anonymität bewusst für eine Verfälschung der Bewertungen im positiven oder negativen Sinne missbraucht wird. Diese Gefahr wird noch durch die Möglichkeit verstärkt, mehrere Pseudonyme zu verwenden.

In beiden Fällen wird die sogenannte Autorschafts-Attribution (kurz AA) relevant, deren Ziel es ist, zu einem anonymen Dokument die Autorschaft zuzuordnen. Ein AA-System verlangt neben dem anonymen Dokument als Eingabe Beispieldokumente von in Frage kommenden Autoren, ein Klassifikationsverfahren sowie stilistische Merkmale, mit de-

ren Hilfe die Autoren voneinander unterschieden werden können. Die Erforschung der Möglichkeiten von AA-Systemen wird in verschiedenen Bereichen betrieben. So strebt die Forensik damit eine Zuordnung von Drohbriefen oder Bekennerschreiben zu bekannten Straftätern an. Für die Literaturwissenschaft ist die Zuordnung anonymer Schriften zu bekannten Schriftstellern eine interessante Herausforderung.

Autoren, die Drohungen hinter dem Schutz der Anonymität aussprechen, muss daher bewusst werden, dass sie durch die AA zunehmend in Bedrängnis geraten. Andererseits ist eine Weiterentwicklung dieser Technologie wichtig, um einen Missbrauch der anonymen Kommentierung einzudämmen, indem zumindest erkannt werden kann, dass eine Person unter mehreren Pseudonymen aktiv ist und so Bewertungen verfälscht.

In unserer Arbeit stellen wir das Konzept der AA vor und beschreiben dabei kurz, wie individueller Schreibstil zu einem durch Software nachweisbaren Merkmal wird. Weiterhin gehen wir der sich daraus ableitenden Frage nach, ob sich Autoren durch Methoden der bewussten Stilverfälschung vor einer Aufdeckung schützen können. Dazu wird zunächst in Kapitel 2 der Begriff *Schreibstil* näher erläutert, bevor in Kapitel 3 AA-Systeme und deren Komponenten beschrieben werden. In Kapitel 4 werden Maßnahmen gegen AA-Systeme präsentiert, wobei menschliche und semi-automatische Täuschungsmethoden diskutiert werden. Abschließend werden in Kapitel 5 weiterführende Möglichkeiten genannt, um vollautomatisierte Täuschungsmethoden realisieren zu können.

2 Schreibstil

Schreibstil stellt in diesem Papier ein wichtiges Konzept dar und wird neben dem Begriff Stil synonym verwendet. Da der Begriff jedoch nicht formalisierbar ist, existieren dafür nach [Gan08, Gol07, Sow73] unterschiedliche Auffassungen. Um Stil dennoch greifbar zu machen, wird daher eine Approximation anhand stilistischer Merkmale benutzt. Diese Merkmale werden im Fachjargon als *Features* bezeichnet und sind zentraler Gegenstand der *Stilometrie*, welche die maßgebende Disziplin der AA darstellt. Vereinfacht ausgedrückt: Stil wird anhand von Features charakterisiert, sodass dadurch Autoren voneinander unterschieden werden können. Wichtigste Randbedingung an Features ist jedoch, dass diese unabhängig von Inhalt (z.B. Fachsprache), Kontext (z.B. angesprochenes Publikum) und Funktion (z.B. sich reimende Wörter in einem Gedicht) sind. Daher gilt es, diese drei Komponenten bei der Approximation des Stils stets auszuschließen.

Features formen den wichtigsten Bestandteil von AA-Systemen. Aufgrund ihrer Vielzahl empfiehlt es sich, diese zu kategorisieren und konzeptionell zu betrachten. Tabelle 1 listet insgesamt 20 solcher Feature-Kategorien auf, die teilweise aus [Hal12] entnommen wurden.

3 Autorschafts-Attributionssysteme

Menschliche Leser können den Stil eines Autors erstaunlich leicht erfassen. Sie benötigen dafür weder eine Vergleichsbasis noch spezielle Verfahren. Meist können sie keine exakte Beschreibung des Stils wiedergeben, doch sie würden ihn in anderen Dokumenten des gleichen Autors wiedererkennen und auch Stilinkonsistenzen, z.B. aufgrund mehrerer Autoren bemerken (zumindest wenn nicht versucht wurde, den Schreibstil aneinander anzupassen). Allerdings haben menschliche Leser auch einige Nachteile. Zu diesen zählen neben Zeit und Kosten auch der Aufwand bzw. die Komplexität bei der Bewältigung der Untersuchung großer Textmengen. Daher wird zunehmend an der automatischen Erfassung von Stil geforscht, wobei Features eine zentrale Rolle einnehmen.

Features werden oftmals dafür kritisiert, Stil nur unvollständig zu charakterisieren. In [Sie13] wird als Begründung die nur schwer mögliche Trennung von Inhalt und Stil durch statistische Verfahren genannt. Weiterer Kritikpunkt ist das fehlende Textverständnis. So können z.B. Argumentation, Humor oder Ironie nur schwer statistisch erfasst werden, da diese auf Verständnis und Hintergrundwissen basieren, die im Text nicht explizit vorkommen. Dennoch wurden Features bereits erfolgreich eingesetzt (vgl. [KG06]). Gute Ergebnisse können erzielt werden, wenn Mensch und Maschine interagieren (menschliche Intuition vereinigt mit der Rechenfähigkeit des Computers). Für aussagekräftige Ergebnisse bietet es sich zudem an, mehrere Features in kombinierter Form zu betrachten, wobei eine solche Kombination als *Feature-Set*) bezeichnet wird. Die drei relevantesten Feature-Sets, die im Verlauf dieser Arbeit eine wichtige Rolle einnehmen, lauten: F_1 = *9 Feature-Set*, F_2 = *Synonym-Based Feature-Set* sowie F_3 = *Writeprint Feature-Set*. Tabelle 2 erläutert, welche Features hierbei konkret enthalten sind.

Neben Features stellen Klassifikatoren ebenfalls einen wichtigen Bestandteil von AA-Systemen dar. Ihre Aufgabe ist es, den wahrscheinlichsten Autor eines anonymen Dokuments anhand der Features und einer Menge von Beispieldokumenten bekannter Autoren vorherzusagen. Dabei gilt die Grundannahme, dass sich der wahre Autor in dieser Menge befindet. Klassifikatoren sind Verfahren aus dem Gebiet des Maschinellen Lernens. In der AA sind die am häufigsten verwendeten Klassifikatoren: Support Vector Machines, k-Nearest Neighbors sowie Naive Bayes. Ein genauerer Überblick über Klassifikatoren findet sich in [Kot07]. Heutzutage existieren bereits einige frei zugängliche AA-Systeme. Eines davon ist *JStylo* (erhältlich unter [PSA13]), welches im Folgekapitel betrachtet wird.

4 Möglichkeiten zur Täuschung der Autorschafts-Attribution

Um angesichts der steigenden Nutzung und Leistung von AA-Systeme anonym schreiben zu können, bedarf es einer gezielten Täuschung. Das gilt besonders dann, wenn andere Texte des Autors bereits öffentlich bekannt sind. Prinzipiell existieren zwei Möglichkeiten zur Täuschung der AA: Die **Anonymisierung** (z.B. durch Löschung diskriminierender Features) und die **Imitation** (z.B. durch Nachahmung von Features anderer Autoren). Als mögliche Gründe für die Täuschung der AA fallen einem zunächst zahlreiche kriminelle

Interessen ein. Es existieren jedoch auch legitime Gründe, warum Autoren an solchen Verfahren interessiert sein könnten. Werden z.B. regimekritische Blogger in totalitären Staatssystemen betrachtet, so ist Anonymisierung ein wesentliches Mittel zum Schutz der Blogger und zu einer wahrheitsgemäßen und dennoch sicheren Berichterstattung in Ländern ohne das Recht zur freien Meinungsäußerung. Wenn also im Folgenden von „Fälscher" die Rede ist, sind damit nicht zwangsläufig kriminell gesinnte Menschen gemeint.

4.1 Manuelle Täuschungsmethoden

Menschen sind in der Lage, Stilbrüche zu erkennen, sodass davon ausgegangen werden kann, dass sie ein Gespür für Stil haben. Daher stellt sich die Frage, ob Menschen mit Hilfe dieses Gespürs ihren eigenen Stil verbergen bzw. einen anderen Stil imitieren können. Gewöhnlich tritt dies nur in Zusammenhang mit kriminellen Absichten auf. Ein häufiges Beispiel sind „falsche" Chat-Freundschaften, die unter anderem von Pädophilen initiiert werden. Legale Imitierung anderer Autoren gibt es zudem in speziellen Wettbewerben, die zu Ehren bekannter Schriftsteller stattfinden und Hobby-Autoren dazu aufrufen, in ihren Beiträgen diese zu imitieren (z.B. *International Imitation Hemingway Competition*). Die Einreichungen zu solchen Wettbewerben enthalten wertvolle Informationen zur menschlichen Stil-Imitation und werden daher auch als Forschungs-Korpora verwendet.

Diese Beispiele finden jedoch Menschen gegenüber statt. Daher ist ein Täuschungserfolg auch davon abhängig, wie leicht sich das Opfer täuschen lässt und hat wenig objektive Aussagekraft über die Fähigkeiten des Fälschers. Aussagekräftiger wäre es zu wissen, wie AA-Systeme auf menschliche Imitation oder Anonymisierung von Texten reagieren. Genau dies wurde an der Drexel University Philadelphia mit dem dort entwickelten Programm *JStylo* getestet, welches die vorgestellten Feature-Sets und verschiedene Klassifikatoren standardmäßig bereitstellt. Dort wurde ein Korpus von Dokumenten mehrerer Hobby-Autoren gebildet (*Brennan-Greenstadt Adversarial Corpus*). Diese sollten von ihnen in der Vergangenheit geschriebene Texte beliebiger Thematik einreichen. Außerdem wurden sie gebeten, einen Text in dem sie ihren eigenen Stil bewusst verstecken, sowie einen Text im Stil des US-amerikanischen Roman-Autors Cormac McCarthy zu schreiben. Sie erhielten dazu einen Auszug aus dessen Roman *The Road*, [McC06]. Die verfälschten (also die anonymisierten und imitierten) Texte hatten im Gegensatz zu den echten Texten vorgegebene Themen: Bei dem anonymisierten Text sollten die Probanden ihre Nachbarschaft beschreiben, bei dem imitierten Text ihren bisherigen Tagesablauf wiedergeben.

JStylo erhielt als Trainingsdaten die echten Texte zugeordnet zum jeweiligen Autor. Für die grundsätzliche Genauigkeit[1] von *JStylo* sollten zunächst erneut echte Texte klassifiziert werden. Dabei wurde auch die Anzahl der Autoren variiert. Als Feature-Sets wurden F_1, F_2 und F_3 verwendet, wobei Letzteres die beste Genauigkeit erzielte (auch bei einer AA mit 40 Autoren lag diese noch bei über 80%). F_2 schnitt auch gut ab, lag jedoch stets ca. 5% unter F_3. Das weniger umfangreiche und weniger komplexe F_1 erreichte dagegen deutlich schlechtere Werte, die ab einer Anzahl von 10 Autoren nur noch unter 50%

[1] Genauigkeit bezeichnet hier den Anteil der korrekt klassifizierten Texte unter allen Klassifikationen.

betrugen. Dennoch sollte bedacht werden, dass F_1 in seiner Genauigkeit weit über einer zufälligen Klassifikation liegt (bei 40 Autoren lag die Genauigkeit von F_1 immerhin bei ca. 25% im Gegensatz zu einer zufälligen Klassifikation mit 2,5%). In Anbetracht der Einfachheit von F_1 ist dies ein gutes Ergebnis.

Mit den verfälschten Texten konnten die Hobby-Autoren, die sich vorher nie mit Stilometrie beschäftigt hatten, *JStylo* sehr stark täuschen. Die Genauigkeit bei der Klassifikation der anonymen Texte anhand von F_3 lag nur noch knapp über der der zufälligen Klassifikation. Die Genauigkeiten von F_1 und F_2 lagen entsprechend darunter und ab einer Zahl von 30 möglichen Autoren sogar nur noch knapp über 0%. Die imitierten Texte trieben sogar die Genauigkeit von F_3 unter die der zufälligen Klassifikation. Für einen Güte-Test der Imitate wurden die Trainingsdaten um CormacMcCarthy selbst als potenziellen Autor sowie Textauszüge aus seinen Romanen ergänzt. *JStylo* sollte anschließend erneut den Autor der Imitate bestimmen. Die höchste Genauigkeit in der Klassifikation von Cormac McCarthy als Autor der imitierten Texte erzielte nur F_1: Es lag bei 5 Autoren knapp unter 70% und bei bis zu 30 Autoren noch über 50%. Ähnliche Werte erreichte F_2. Dagegen erzielte F_3 im Schnitt nur halb so hohe Werte wie die anderen beiden. Detaillierte Ergebnisse finden sich in [BAG12]. Der verwendete Korpus dagegen ist unter [PSA13] erhältlich.

Diese Studie zeigt zum einen, dass Menschen ihren Stil auch objektiv messbar verschleiern und dadurch AA-Systeme täuschen können. Die Teilnehmer konnten ihre Texte *JStylo* gegenüber erfolgreich anonymisieren. Dabei war die Imitation eines anderen Stils erfolgreicher als die Anonymisierung durch lediglich Verbergen des eigenen Stils. Zum anderen zeigt die Studie, dass Stilnachahmung funktioniert. Bei genauerer Betrachtung der Ergebnisse fällt jedoch folgendes auf: Vergleicht man die Schaubilder der AA der echten Texte und der AA der imitierten Texte unter Hinzunahme von Cormac McCarthy als potenziellen Autor, so hat sich die Platzierung der einzelnen Feature-Sets bezüglich ihrer erzielten Genauigkeit gerade umgekehrt. Da eine hohe Genauigkeit bei der Zuordnung zu Cormac McCarthy aber gerade eine hohe Täuschungsanfälligkeit des Feature-Sets bedeutet, ergibt dieser Unterschied wieder Sinn: Er unterstreicht die Qualität von F_3 zur besseren Resistenz gegen Täuschungsversuche. Diese guten Eigenschaften sind vor allem auf die Komplexität von F_3 zurückzuführen. So können die Features darin (z.B. einzelne Buchstabenhäufigkeiten) kaum gegenüber einfacheren Features in F_1 (z.B. durchschnittliche Satzlänge) beeinflusst werden. Diese Auffälligkeiten liefern Forschern wiederum Anhaltspunkte, um die Täuschung der AA zu erkennen. Diesem Thema widmet sich z.B. [ABG12]. Allerdings bedeutet das Erkennen einer Täuschung noch lange nicht die Identifikation des wahren Autors. Eine Anonymisierung durch das intuitive Verbergen oder Verändern von Stil ist also in der Regel gewährleistet.

4.1.1 Wie verändern Menschen ihren Stil

Menschen können eher die Werte der Features in F_1 verändern als die in F_3. Welche Features es genau sind, wurde von den Forschern in [BAG12] ebenfalls herausgearbeitet. Zunächst sagten die Probanden selbst, sie verwendeten bei der Anonymisierung eher kürzere und einfachere Sätze und bei der Imitation von Cormac McCarthy eine beschreibendere und düsterere Sprache. Der statistische Vergleich der einzelnen Feature-Werte

in echten und verfälschten Texten unterstützt diese Aussagen zum Teil: Demnach gab es bei der Anonymisierung vor allem Abnahmen der durchschnittlichen Wort-/Satzlänge und Silbenzahl, aber auch eine Zunahme von Adverbien. Bei der Imitation gab es ebenfalls Abnahmen der durchschnittlichen Wort-/Satzlänge und Silbenzahl, sowie weitere Abnahmen von Adjektiven und Adverben und eine Zunahme von Funktionswörtern. Insgesamt schließen die Forscher, dass sich die Komplexität von verfälschten Texten verringert. Diese Ergebnisse sind mit dem Hinweis zu interpretieren, dass die Thematik der echten Texte nicht vorgegeben war, jedoch die anonymisierten und die imitierten Texte jeweils ein vorgegebenes Thema hatten (Beschreibung der Nachbarschaft, Beschreibung des bisherigen Tagesablaufs). Hier war also die geforderte Trennung des Stils von Inhalt, Kontext und Funktion nicht gewährleistet: Alle gefälschten Texte hatten eine beschreibende Funktion, was die Zunahme von Adverben und Adjektiven aus der Funktion heraus begründet. Bei den Imitaten spielt weiterhin der zu imitierende Autor (also der Kontext) eine wichtige Rolle. Hier wären weitere Studien mit themenunabhängigen Korpora nötig, um allgemeingültige Ergebnisse zu erhalten.

4.1.2 Schwierigkeiten und Grenzen

Die Studie in [BAG12] und viele alltägliche Beispiele zeigen, dass sich Menschen ihres persönlichen Stils durchaus bewusst und darüber hinaus in der Lage sind, diesen zu verbergen oder zu verändern. Sie führen damit in erster Linie andere Menschen in die Irre, können jedoch auch eine Anonymisierung im Hinblick auf Computerprogramme erzielen. Die Imitation eines anderen Autors gelingt ihnen nur in Bezug auf einfache Features gut. AA-Systeme können jedoch solche Imitate durch Benutzung komplexerer Feature-Sets entlarven (z.B. einzelne Buchstabenhäufigkeiten). Trotz alledem bleibt der Autor des Imitats anonym. Gerade bei sehr langen oder einer großen Anzahl von Texten können jedoch Schwierigkeiten auftreten: Einen falschen Stil können nur die wenigsten über längere Zeit konsistent einhalten. Zudem ist das Imitieren bzw. Unterdrücken von Stil äußerst anstrengend. Schwieriger wird es noch, wenn existierende Texte nachträglich anonymisiert werden sollen, ohne dabei die Semantik zu verändern.

4.2 Semi-automatische Täuschungsmethoden

Die Grenzen der menschlichen Täuschungsmethoden wecken den Wunsch nach computergestützter Anonymisierung und Stilverfremdung. Programme zur Anonymisierung von Texten müssen nun noch einen Schritt weiter gehen, sodass sie den Text nach erfolgreicher Bearbeitung nicht mehr zuverlässig klassifizieren können. Ein solches Programm wird im weiteren Verlauf vorgestellt. Zunächst werden jedoch zwei weitere Ansätze erläutert.

4.2.1 Übersetzung und Rück-Übersetzung

Schon im Jahr 2000 entstand die Idee, computergestützte Anonymisierung durch maschinelles Übersetzen in eine andere Sprache und Rück-Übersetzung in die Ausgangssprache

zu realisieren. Solche Übersetzungsdienste sind frei verfügbar (z.B. *Google Translate* oder *Bing Translator*) und einfach zu bedienen. Allerdings haben sie einen eher schlechten Ruf bezüglich der Qualität ihrer durchgeführten Übersetzungen. Daher gilt es zu prüfen, ob die Semantik beibehalten wird und falls ja, ob die Schreibstile ausreichend verändert werden können. Eine Studie über die Auswirkung von Übersetzungen auf die AA wurde in [CG12] durchgeführt. Hier wird zunächst herausgestellt, dass maschinelle Übersetzer Spuren in Texten hinterlassen (*Translator-Effect*), woran sogar verschiedene Übersetzer erkannt werden können. Der Übersetzer wird also wie ein zweiter Autor behandelt, der seinerseits Features im Text hinterlässt. Es stellt sich die Frage, ob diese die Features der menschlichen Autoren verstärken, abschwächen oder ob beide Merkmale ungestört in einem Text koexistieren können. Für die Studie wurden Texte[2] den folgenden Übersetzungsfolgen unterzogen: (`en` $\rightarrow$ `de` $\rightarrow$ `en`), (`en` $\rightarrow$ `ja` $\rightarrow$ `en`), sowie (`en` $\rightarrow$ `ja` $\rightarrow$ `de` $\rightarrow$ `en`), mit `en` = Englisch, `de` = Deutsch und `ja` = Japanisch. Als AA-System diente erneut *JStylo* unter Zuhilfenahme von *Google Translate* und *Bing Translator*[3].

Verschiedene Features wurden auf den übersetzten Texten getestet und aus den besten ein eigenes Feature-Set F_4 = **Translation Feature-Set** zusammengestellt. Dieses beinhaltet die erfolgreichsten Features für die Bestimmung des Übersetzers sowie für die AA. Dazu gehören unter anderem Buchstaben Bi- und Trigramme, Wortlänge, Zeichensetzung und Funktionswörter. Eine Studie auf dem gleichen Korpus mit den gleichen Übersetzern, aber den von *JStylo* standardmäßig angebotenen Feature-Sets ist in [BAG12] zu finden. Detaillierte Ergebnisse der AA seitens *JStylo* und die Relevanz einzelner Features aus F_4 finden sich in [CG12]. Auffällig ist unter anderem der große Unterschied der Relevanz von Funktionswörtern bezüglich der AA und der Bestimmung des Übersetzers. Während fürs Letztere der Wert dieses Features sehr relevant ist, scheint es für die AA nur eine kleinere Rolle zu spielen. Als die Funktionswörter jedoch testweise für die AA aus F_4 ausgeschlossen wurden, verschlechterte sich die Genauigkeit im Schnitt um ca. 15%. Daraus schließen die Forscher, dass solche eher übersetzerspezifischen Features, die durch den Translator-Effect in den Text gelangen, die AA nicht bedeutend stören, sondern im Gegenteil sogar unterstützen können.

Insgesamt zeigen die Ergebnisse eine hohe Genauigkeit bezüglich der AA von durchschnittlich 91,54%. Die Ausgangstexte wurden im Schnitt mit 97,62% klassifiziert, was eine Reduktion durch die Übersetzungsvorgänge von ca. 6% bedeutet. Die Genauigkeit reicht dabei von ca. 77% für die Übersetzung mit zwei Zwischensprachen durch Bing bis hin zu 100% mit der Zwischensprache Japanisch und dem gleichen Übersetzer. Die wichtigste Erkenntnis ist, dass die Übersetzungsvorgänge keine Anonymisierung der Ausgangstexte erreichen. Dadurch, dass die niedrigsten Genauigkeiten mit zwei Zwischensprachen erreicht werden konnten (77,78% durch Google und 87,3% durch Bing), ergibt sich die Vermutung, dass die Hinzunahme weiterer Zwischensprachen die AA ungenauer werden lässt. Davor sollte jedoch ein Blick auf die Sinnerhaltung der bereits getesteten Übersetzungsfolgen geworfen werden. In [BAG12] werden hierzu einige Beispielübersetzungen gegeben. Zunächst eine schlechte Übersetzung am Beispiel: *„Just remember that the things you put into your head are there forever, he said.“*, [McC06]:

[2] Hier wurde der *Brennan-Greenstadt Korpus* ebenfalls verwendet (jedoch ohne die verfälschten Texte).

[3] Beide nutzen das gleiche Übersetzungsverfahren, unterscheiden sich jedoch in ihren Übersetzungen.

en → de → en	*„Remember that the things that you are dead set on always there, he said.“*
en → ja → en	*„But things are there forever remember what you put in your head, he said.“*
en → de → jp → en	*„You are dead, that there always is set, please do not forget what he said.“*

Nun eine gute Übersetzung, aber keine Stil-Veränderung anhand des folgenden Beispielsatzes: *„They passed through the city at noon of the day following.“*, [McC06]:

en → de → en	*„They crossed the city at noon the following day.“*
en → ja → en	*„They passed the city at noon the following day.“*
en → de → ja → en	*„They crossed the city at noon the next day.“*

Beide Studien bewerteten das Täuschungsverfahren als insgesamt ungeeignet, da zum einem die Anonymisierung nicht stark genug ist und zum anderen die Semantik des Textes verfälscht wird. In [BAG12] wird hinzugefügt, dass es durchaus gut übersetzte und anonymisierte Sätze in den Übersetzungen gegeben habe, aber dass diese von im Hinblick auf Sinn oder Grad der Anonymisierung unzureichenden Sätzen dominiert wurden und so das Gesamtergebnis unbrauchbar sei. Auch die Hinzunahme weiterer Zwischensprachen scheint angesichts der Sinnverfälschung nicht vielversprechend. Fortschritte im Bereich der maschinellen Übersetzung könnten jedoch eine Verbesserung des Verfahrens zukünftig ermöglichen. Unklar ist, welche Rolle die gewählten Sprachen für das Verfahren spielen. Mit Deutsch wurde eine dem Englischen ähnliche und mit Japanisch eine vollkommen unähnliche Sprache verwendet.

4.2.2 Eliminierung typischer Wörter

Kacmarcik und Gamon forschten in [KG06] an den *Federalist Papers*. Hierbei handelt es sich um eine Kollektion von 85 Artikeln, die 1788 in den USA anonym veröffentlicht wurden. Mittlerweile sind die Autorschaften der meisten Texte eindeutig geklärt. Demnach sind 5 Artikel von John Jay, 51 von Alexander Hamilton, 14 von John Madison und 3 Artikel wurden von Madison und Hamilton gemeinsam geschrieben. Die Autorschaft der verbleibenden 12 Dokumente ist nicht eindeutig geklärt. 1964 führten jedoch Stil-Analysen der 12 Texte zu der Annahme, John Madison sei ihr Verfasser. [4].

In ihrer Studie testeten Kacmarcik und Gamon das systematische Verfälschen bestimmter Features in diesen 12 Texten. Ziel war dabei, die Autorschaft Madisons mit den gleichen stilometrischen Methoden wie von 1964 nicht mehr nachweisbar zu machen und stattdessen die Texte Hamilton zuzuordnen. Die Forscher konzentrierten sich dabei auf die Angleichung der Anzahl von „unterscheidenden“ Wörtern in den 12 zu klassifizierenden Texten an die Anzahl dieser Wörter in Hamiltons Texten. Sie entwickelten einen Algorithmus (aufgeführt in [KG06]), der unterscheidende Wörter identifiziert und den Fälscher anweist,

[4] Die bisher populärste Errungenschaft auf dem Gebiet der AA.

wie bestimmte Worthäufigkeiten zu ändern sind, um eine Zuordnung zu Hamiltons Texten zu erreichen. Leider erzielten sie mit der Anpassung der 10 am meisten unterscheidenden Wörter nicht die gewünschte Anonymisierung. Als Grund dafür fanden sie heraus, dass diese insgesamt zu selten vorkamen, als dass sie einen großen Unterschied hätten erzeugen können. Darum erfolgte der nächste Versuch mit den 10 am meisten unterscheidenden Worten, die zusätzlich eine bestimmte Häufigkeit im Text erreichten. Mit dieser Methode gelang es ihnen, alle 12 Artikel so verändern, dass sie durch das gleiche Verfahren wie von 1964 Hamilton zugeordnet wurden. Zur erfolgreichen Manipulation der Texte benötigten Kacmarcik und Gamon pro 1000 Worte durchschnittlich 14,2 Veränderungen. Sie erreichten damit im Schnitt eine Reduzierung der Wahrscheinlichkeit der Autorschaft Madisons von 96,93% auf nur 12,51%. Kacmarcik und Gamon betonen jedoch, dass ihr Verfahren nur eine „seichte Anonymisierung“ ermöglicht hat, welche komplexeren Feature-Sets leider nicht standhält. Dennoch konnten sie in [KG06] zeigen, dass stilometrische Faktoren computergestützt beeinflusst werden können. Sie halten die Weiterentwicklung dieser Techniken zu umfangreicheren Programmen zur Täuschung der AA für realistisch.

4.2.3 Anonymouth

Anonymouth (erhältlich unter [PSA13]) ist ein Programm zur Anonymisierung von Texten. Es wurde ebenfalls an der Drexel University Philadelphia entwickelt und ist in [MAC+12] ausführlich beschrieben. *Anonymouth* interagiert mit *JStylo* und hat das Ziel, diesen zu täuschen. Als Input werden dabei das zu verfälschende Dokument $\mathcal{D}$, andere Dokumente des gleichen Autors $\mathbb{D}_{same}$ sowie Dokumente anderer Autoren $\mathbb{D}_{others}$ verlangt. Anschließend müssen verschiedene Feature-Sets ausgewählt werden, anhand derer $\mathcal{D}$ analysiert wird. Dabei legen $\mathbb{D}_{others}$ fest, wie die Werte der Features sein sollten und $\mathbb{D}_{same}$ wie sie möglichst nicht sein sollten. *Anonymouth* gibt dann für jedes Feature den Ist- und Soll-Wert an. Für komplexe Feature-Sets findet eine Priorisierung der Features statt, d.h. es werden z.B. nur die 5 Features angezeigt, deren Ist-Wert am meisten vom Soll-Wert abweicht. Darüber hinaus werden die Stellen visualisiert, an denen das Feature in $\mathcal{D}$ auftritt und der Nutzer angewiesen, an einigen dieser Stellen Änderungen vorzunehmen. Anschließend wird $\mathcal{D}$ erneut analysiert. Der Prozess wiederholt sich solange, bis der Nutzer ihn abbricht oder die gewünschte Genauigkeit erreicht ist. In jeder Iteration bekommt der Nutzer das Ergebnis der AA angezeigt. Diese schrittweise Anonymisierung ist notwendig, da sich viele Features gegenseitig beeinflussen (Anzahl der Sätze, durchschnittliche Satzlänge, etc.) und $\mathcal{D}$ daher stets re-klassifiziert werden muss. Die Anonymisierung gilt als erfolgreich, falls der wahre Autor anhand des gewählten Feature-Sets nur noch mit einer kleineren Wahrscheinlichkeit als bei einer zufälligen Zuordnung seitens *JStylo* vorhergesagt werden kann.

Die Entwickler selbst bezeichnen *Anonymouth* nur als ersten Schritt in die Richtung computergestützter Anonymisierung von Texten. *Anonymouth* nimmt wie bereits erwähnt keine vollautomatische Anonymisierung vor, sondern gibt nur Anweisungen. Das Ändern von $\mathcal{D}$ geschieht weiterhin manuell durch den Nutzer, wodurch sich einige Schwächen ergeben. So können dem Nutzer bestimmte Änderungen nicht zugemutet werden (z.B. das Ändern von n-Grammen oder einzelnen Buchstabenhäufigkeiten, vgl. [MAC+12]). Wei-

terhin kann die gegenseitige Beeinflussung vieler Features und die dadurch verbundene iterative Anonymisierung den Vorgang sehr in die Länge ziehen. Eine Benutzbarkeitsstudie in [MAC+12] ergab, dass einige Nutzer ihren Text in 30-60 Minuten anonymisieren konnten, während anderen Nutzern die auf eine Stunde begrenzte Zeit jedoch nicht reichte, um den gewünschten Grad der Anonymisierung zu erzielen. Außerdem verlangt *Anonymouth* vom Nutzer einen Korpus, die Auswahl von Feature-Sets und ein Klassifikator, was wiederum Hintergrundwissen voraussetzt.

5 Zusammenfassung und Ausblick

Die Weiterentwicklung von AA-Systemen ermöglicht zunehmend, öffentlich zugängliche vermeintlich anonyme Texte ihren Autoren zuzuordnen. Dies kann eine Beeinträchtigung der Privatsphäre der Autoren zur Folge haben. Aus diesem Grund wird zunehmend mehr an der Täuschung dieser Systeme geforscht. Verschiedene Ansätze und Studien zu diesem neuen Forschungsgebiet wurden in dieser Arbeit vorgestellt. Die beschriebenen Grenzen haben aufgezeigt, dass menschliche Täuschungsmethoden impraktikabel erscheinen. Dagegen bieten semi-automatische Methoden bessere Ergebnisse im Hinblick auf die Täuschung von AA-Systemen. Aber auch hier zeigen sich Grenzen. Sollen z.B. viele Texte (eventuell sogar simultan) anonymisiert werden, so sind diese Methoden ungeeignet, da hier immer noch der Mensch die Stiländerung selbst durchführen müsste. Vollautomatisierte Täuschungsmethoden stellen eine Alternative dar, die diese Problematik beheben und weitere Einschränkungen aufheben würde. Sie können Stiländerung ohne dem Benutzer durchführen und setzen kein linguistisches Hintergrundwissen voraussetzen.

Eine Möglichkeit zur Realisierung vollautomatisierter Täuschungsmethoden sind Natural Language Watermarking Verfahren, welche beispielsweise in [HSWZ13, Klo14] vorgeschlagen werden. Zwar verfolgen diese einen anderen Zweck (Einbettung verdeckter Nachrichten), ermöglichen jedoch automatisierte Textumformungen, die auf allen sprachlichen Ebenen (phonemisch, morphologisch, lexikalisch, syntaktisch aber auch semantisch) Änderungen durchführen. Diese Änderungen haben stilistische Verzerrungen zur Folge, was wiederum verhelfen kann, AA-Systeme zu täuschen. Oberste Priorität jedoch ist, die Semantik des Textes möglichst weitgehend zu erhalten. Andernfalls wäre der Text nach den Umformungen unlesbar. Im Rahmen zweier Studien zeigte sich in [HSWZ13] mit 89 bzw. in [Klo14] mit 42 Teilnehmern, dass deren vorgeschlagene Natural Language Watermarking Verfahren die Semantik deutschsprachiger Ausgangstexte nur geringfügig verzerrten, sodass die Mehrheit der Teilnehmer die Umformungen nicht signifikant wahrnehmen konnten. In Zukunft gilt es, die Tauglichkeit dieser Verfahren zum Zwecke der Anonymisierung des persönlichen Schreibstils zu analysieren in der Hoffnung, AA-Systeme dadurch erfolgreich täuschen zu können.

6 Danksagung

Diese Arbeit wurde unterstützt vom CASED - Center for Advanced Security Research Darmstadt (www.cased.de), gefördert vom Hessischen Ministerium für Wissenschaft und Kunst unter dem LOEWE-Förderprogramm.

Literatur

[ABG12] S. Afroz, M. Brennan und R. Greenstadt. Detecting Hoaxes, Frauds, and Deception in Writing Style Online. In *Security and Privacy (SP), 2012 IEEE Symposium on*, Seiten 461–475, 2012.

[BAG12] Michael Brennan, Sadia Afroz und Rachel Greenstadt. Adversarial Stylometry: Circumventing Authorship Recognition to Preserve Privacy and Anonymity. *ACM Trans. Inf. Syst. Secur.*, 15(3):12:1–12:22, November 2012.

[CG12] A. Caliskan und R. Greenstadt. Translate Once, Translate Twice, Translate Thrice and Attribute: Identifying Authors and Machine Translation Tools in Translated Text. In *2012 IEEE Sixth International Conference on Semantic Computing (ICSC)*, Seiten 121–125, 2012.

[CH07] Jonathan H. Clark und Charles J. Hannon. A Classifier System for Author Recognition Using Synonym-Based Features. In *Proceedings of the artificial intelligence 6th Mexican international conference on Advances in artificial intelligence*, MICAI'07, Seiten 839–849, Berlin, Heidelberg, 2007. Springer-Verlag.

[Gan08] Prof. Dr. Christina Gansel. Philologische Methoden, Ernst-Moritz-Arndt-Universität Greifswald Philosophische Fakultät (Institut fr Deutsche Philologie), 2008. Letzter Zugriff: 13.02.2014.

[Gol07] Felix Golcher. Ein Einblick in die statistische Stilometrie, 2007.

[Hal12] Oren Halvani. Autorenschaftsanalyse im Kontext der Attributation, Verifikation und intrinsischen Exploration. Master thesis, Technische Universität Darmstadt, 2012.

[HSWZ13] Oren Halvani, Martin Steinebach, Patrick Wolf und Ralf Zimmermann. Natural Language Watermarking for German Texts. In ACM, Hrsg., *Proceedings of the 1st ACM Workshop on Information Hiding and Multimedia Security, June 17-19, 2013 Montpellier, France*, 2013.

[KG06] Gary Kacmarcik und Michael Gamon. Obfuscating Document Stylometry to Preserve Author Anonymity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, Seiten 444–451, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[Klo14] Peter Kloeckner. *Phonemische, Lexikalische und Syntaktische Natural-Language-Watermarking-Verfahren*. Bachelor thesis, Technische Universität Darmstadt, 2014.

[Kot07] S. B. Kotsiantis. Supervised Machine Learning: A Review of Classification Techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, Seiten 3–24, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press.

[MAC+12] AndrewW.E. McDonald, Sadia Afroz, Aylin Caliskan, Ariel Stolerman und Rachel Greenstadt. Use Fewer Instances of the Letter ï": Toward Writing Style Anonymization. In Simone Fischer-Hbner und Matthew Wright, Hrsg., *Privacy Enhancing Technologies*, Jgg. 7384 of *Lecture Notes in Computer Science*, Seiten 299–318. Springer Berlin Heidelberg, 2012.

[MB11] Rachel Greenstadt Michael Brennan, Sadia Afroz. Deceiving Autorship Attribution. Bericht, Drexel University Philadelphia, 2011.

[McC06] C. McCarthy. *The Road*. Oprah's Book Club. Vintage Books, 2006.

[PSA13] PSAL. Drexel University's Privacy, Security, and Automation Laboratory. JStylo-Anonymouth Webseite: https://psal.cs.drexel.edu/index.php/JStylo-Anonymouth, 2013. Letzter Zugriff: 13.02.2014.

[Sie13] Martin Siefkes. Stil und Gesellschaft - Plädoyer für eine allgemeine Stilistik. 2013.

[Sow73] Bernhard Sowinski. *Deutsche Stilistik*. Fischer-Taschenbücher. Fischer Taschenbuch Verlag, 1973.

7 Anhang

Feature-Kategorie	**Kurzbeschreibung / Beispiele**
Interpunktionszeichen	(,), [,], !, ?, ;, :, . . .
Buchstaben	A-Z, Ä, Ö, Ü, a-z, ä, ö, ü, ß
Buchstaben n-Gramme	Textbeispiel $\xrightarrow{n=2}$ {Te, ex, xt, tb, be, . . .}, $\xrightarrow{n=3}$ {Tex, ext, xtb, tbe, . . .}, . . .
Präfixe	**Text**beispiel (Vorsilbe)
Infixe	Text**bei**spiel (innerer Wortbestandteil)
Suffixe	Textbei**spiel** (Nachsilbe)
Funktionswörter	Artikel (der, das, einer, eines, . . .), Konjunktionen (und, oder, . . .), . . .
Anglizismen	Wortentlehnungen (z.B. Mail, Newsletter, Chat, Meeting, Update, . . .)
Neologismen	Kunstwörter (z.B. Abmahnwelle, Nerd, googeln, verschlimmbessern, . . .)
Wort n-Gramme	Ein kleines Textbeispiel $\xrightarrow{n=2}$ {(Ein kleines), (kleines Textbeispiel)}
Kollokationen	Häufig vorkommende Wortverbindungen (z.B. *starker Tobak*)
Wortarten	Adjektive, Interjektion, Numerale, Substantive, . . .
Wortart n-Gramme	(Artikel-Adjektiv-Nomen), (Pronomen-Nomen-Artikel), . . .
Phrasen/Redewendungen	Redensarten (z.B. *aus dem Nähkästchen plaudern*)
Satz-Anfänge/Endungen	Satzanfang(Nomen), Satzende(finites Verb), . . .
Wort-Komplexität	Wörter bestimmter Länge, Wörter mit x Vokalen
Satz-Komplexität	Sätze bestimmter Länge, Vorfeld/Mittelfeld/Nachfeld-Komplexitäten, . . .
Text-Komplexität	Funktionswort-Dichte, Koreferenzketten, . . .
Verständlichkeits-Indizes	*Gunning Fog Readability Index*, *Flesch-Kincaid Reading Ease*, . . .
Grammatikalische Fehler	Falsche Verwendung von Genus, Kasus, Kommata, . . .

Tabelle 1: Eine Auswahl von 20 Feature-Kategorien, teilweise aus [Hal12] entnommen.

Feature-Set	**Enthaltene Features**
F_1 = *9 Feature-Set*	Enthält unter anderem x = die Anzahl nur einmal vorkommender Wörter, Verhältnis von x zur Anzahl aller Wörter, durchschnittliche Silbenzahl pro Wort, durchschnittliche Satzlänge, Anzahl von Zeichen, Buchstaben und Sätze sowie *Gunning Fog Readability Index* und *Flesch-Kincaid Readability Ease*, [BAG12].
F_2 = *Synonym-Based Feature-Set*	Enthält die Anzahl der möglichen Synonymen von Wörtern. Je mehr Synonyme existieren, desto charakteristischer ist das Wort für den Stil. Berücksichtigt wird dabei außerdem die jeweilige Worthäufigkeit im zu testenden Text und in der Vergleichsbasis, [CH07].
F_3 = *Writeprint Feature-Set*	Häufigkeiten von spezifischen Zeichen/Symbolen, Interpunktionszeichen, Ziffern, Buchstaben, Wörtern, Funktionswörtern sowie Wortarten. Weiterhin: Anzahl aller Zeichen, kurzer Worte sowie aller Wörter. Prozentualer Anteil von Ziffern, großgeschriebenen Buchstaben sowie gängigen Zeichen Bi- und Trigrammen. Verhältnis von Hapax und Dis legomena (Wörter, die nur einmal bzw. zweimal in einem Text vorkommen), sowie durchschnittliche Wortlänge, [BAG12].

Tabelle 2: Die drei relevantesten Feature-Sets im Rahmen dieser Arbeit.

Recent Developments in Covert Acoustical Communications

Michael Hanspach and Michael Goetz
Fraunhofer FKIE, Wachtberg, Germany
{michael.hanspach, michael.goetz}@fkie.fraunhofer.de

Abstract: The topic of covert acoustical communication in air has produced public attention, lately. In 2013, we presented the fact that infected drones (e.g. laptops) might form covert acoustical mesh networks in air, only utilizing the built-in speakers and microphones. In this article, we aim to present the current state of covert acoustical communication in air. We discuss future forms of covert acoustical mesh networks, we present an alternative audio signal that is improved in terms of stealthiness, and we provide considerations on the stealthiness of covert acoustical communication in air. In order to protect computing setups from malware utilizing covert acoustical communications, we present an analysis on detection and localization mechanisms for covert acoustical transmissions. Finally, we describe the implementation of an audio intrusion detection system that is designed for automatic analysis of acoustical emanations.

1 Introduction

Covert acoustical communication in air is a serious threat to computer security as identified by Hanspach, Keller and Goetz [HK13b, HK13a, HG13], as it may circumvent operating system and network security policies, unless this type of attack is considered in the security design of the computing system's components.

In November 2013, we finally published the concept and implementation of a covert acoustical mesh network in air [HG13]. Reutilizing the Adaptive Communication System (ACS) (see also [ON10, NG12, GN12, MTZ10]) that has originally been developed for robust acoustic underwater communication by the Research Department for Underwater Acoustics and Marine Geophysics (FWG) of the Bundeswehr Technical Center WTD 71, Kiel, Germany, a robust low-bandwidth communication scheme for air-based wireless mesh networks was discussed. Following the presented approach, future malware might utilize covert acoustical communications to even extract information from computers that are not connected to any regular type of network (e.g. Ethernet or WLAN). Our work on covert acoustical mesh networks has been widely covered by public media (e.g. [Goo13, Owa13, Com13]).

Further research on these covert networks is important, as current security policies might be easily circumvented by covert communications. This threat is even intensified by the

Corresponding author email: michael.hanspach@fkie.fraunhofer.de

fact that some devices (e.g. smartphones) are not shipped without speakers and microphones, and a mobile phone without microphones and speakers would be useless in regard to its original purpose (i.e. phone conversations).

In the terminology of this article, the term *covert* refers to the fact that we are using means for communication that have not been designed for computer-to-computer communication at all (i.e. acoustic wave propagation with built-in audio devices). Because of this, the presented acoustical channel can be described as a *covert channel* [Lam73]. Applying the definition of a covert channel to the context of computing systems separated by air gaps [IET07], we define a **covert network** as:

A network that exists independently from established types of network interfaces and utilizes physical means for computer-to-computer communication that have not been designed for this type of communication.

In difference to this *covertness*, the term *stealthiness* refers to the fact that we hide the very presence of communication from detection of human computer users. Stealthiness is *not* an absolute property. Instead, different degrees of stealthiness can be conceived as distinct points of a stealthiness continuum. Increasing the degree of stealthiness of a covert network is only possible to a certain degree and can sometimes come with drawbacks regarding the impact of the covert network (e.g. reducing the maximum range per hop as described in Section 2).

Our contribution to the field includes the presentation of an alternative audio signal, considerations on the stealthiness of a covert acoustical malware, the presentation of detection and localization mechanisms for covert acoustical communications, and the implementation of an audio intrusion detection system.

The remainder of this article is structured as follows. Details on the implementation and evaluation of an alternative audio signal within the ultrasonic frequency range and further considerations on stealthiness are provided in Section 2. In Section 3, we present methods for detecting covert acoustical communications in the presented form. In Section 4, we present the implementation of an adaptive audio intrusion detection system. In Section 5, we briefly discuss related work and how it differs from our work. In Section 6, we conclude the article.

2 Advances in covert acoustical communications

2.1 Implementation and evaluation of an alternative signal that is placed within the ultrasonic frequency range

To demonstrate the high transmission range per hop achievable (19.7 m) within this covert network, the audio transmissions generated in the first experiment on covert acoustical communications [HG13] have been placed into the near ultrasonic frequency range $<$ 20,000 Hz. Although the sound processor of commonly available computers (i.e. Lenovo T400 laptops) gradually attenuates the ultrasonic frequency range between 20,000 Hz and

25,000 Hz [HG13], audio transmissions might also be placed into a narrow band of the available ultrasonic frequency range $\geq$ 20,000 Hz in order to prevent any chance of human detection (even at high volume levels). However, as an acoustical malware would be in control of the volume levels of any modem transmission, and might adapt to the configured system volume levels, the original signal already delivers a relatively high degree of stealthiness and might not be detected easily.

The remainder of this section includes an alternative approach where the audio signal of the ACS modem is put entirely into the ultrasonic frequency range. The alternative audio signal implements FHSS (frequency hopping spread spectrum) with 20 carriers, providing an ultrasonic frequency range from 20,500 to 21,500 Hz that may fit just right into the available ultrasonic frequency range of the utilized sound processor. The bit rate of approximately 20 bit/s and the latency of 6 s per hop remain unchanged with this frequency shift. Although the delivered bandwidth is not comparable to the capabilities of the established wireless communication standards, it is actually high enough to transfer small portions of critical data such as keystrokes from the keyboard of a human computer user, private encryption keys and malicious command-and-control data [HG13].

Moreover, much higher transmission rates are achievable at a closer distance, as we will show in a future article.

While humans walking through the experiment setup are found to be interfering with the transmissions, this effect does not defeat the effectiveness of the covert network at all, as GUWMANET/GUWAL transmissions without a received acknowledgment are simply retransmitted [GN12]. The frequency range of the alternative audio signal is shown in Fig. 1.

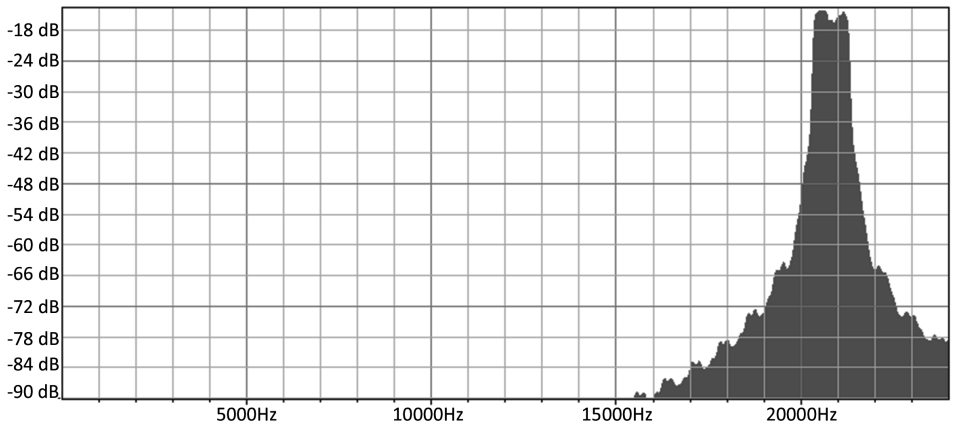


Figure 1: Frequency range (before filtering) of the alternative signal with the ACS modem (FFT, size 1024, Hanning window)

As can be seen in Fig. 1, the frequency range has been completely moved into the ultrasonic frequency range $\geq$ 20,000 Hz at relative volume levels between -12 dB and -48 dB. The spectrogram of the alternative audio signal is shown in Fig. 2.

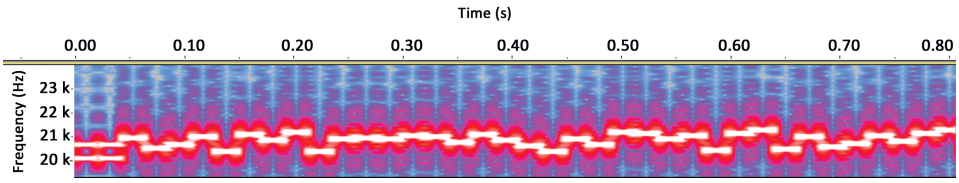


Figure 2: Spectrogram (before filtering) of the alternative signal with the ACS modem

The waveform (Fig. 3) was adapted in order to prevent intersymbol interference by application of a trapezoid window and the square root of a Hamming window defined by: $\sqrt{0.54 + 0.46 \cdot \cos\left(\frac{2\pi(i-1024/2)}{1024}\right)}$ for each index i. These measures have shown to produce less acoustic energy in the surrounding frequencies.

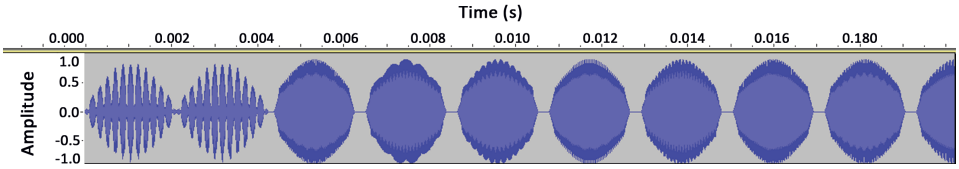


Figure 3: The adapted waveform (windows applied)

To support audio transmissions with the alternative signal, and to increase the stealthiness at high volume levels, a newly adapted bandpass FIR filter (Fig. 4) is utilized for the main part of the signal.

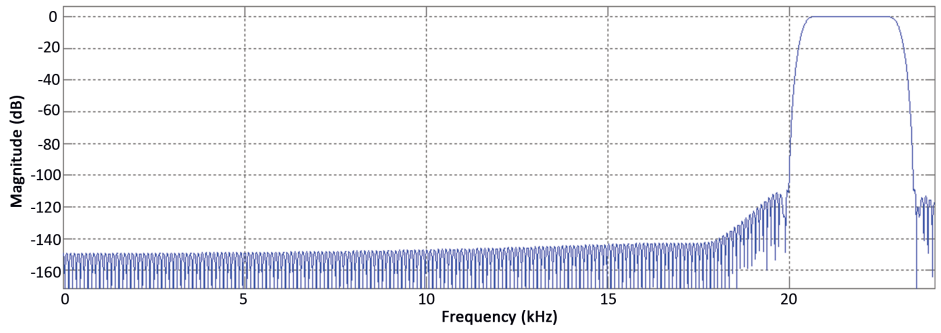


Figure 4: The newly designed bandpass FIR filter (Blackman-Harris window, 20.4 kHz - 23.0 kHz)

In order to prevent potentially audible artifacts of the signal, the bandpass FIR filter is installed both at the receiver and the transmitter. Because of the audio filtering, it is even possible to play mp3 files at the same time as transmitting data.

2.2 Range Experiment

The range experiment (see also [HG13]) is rerun in order to determine the maximum range achievable per hop. The range experiment is described in the following. Two isolated computing systems (type: Lenovo T400) are placed in an approximately 25 m long corridor. The laptop covers are opened and the laptop audio input and output devices are loosely directed at each other. No acoustical preparations are made in the course of the experiment setup. The maximum transmission range between 2 nodes is determined by sending out messages, and by measuring and gradually increasing the transmission range between the nodes with each successful transmission.

As a result of the range experiment, it was determined that a maximum range of 8.2 m per hop can be achieved with the alternative signal (19.7 m with the originally presented signal). Therefore, a potential attacker would have to choose between maximum stealthiness and maximum communication range in the design of a malware.

Further tests with a prerecorded and replayed audio transmission suggest that other types of devices (i.e. smartphones) are, in fact, able to communicate at a comparable distance.

2.3 Considerations on stealthiness

With the alternative signal, a *covert network* might be *stealthy* in the following meanings:

1. A covert network might be *stealthy* by running the communication system as a background process on the computing system and might implement advanced process hiding mechanisms (see also [Ger04, Flo05]).
2. A covert network might be operating in a very *stealthy* state, i.e. only occasionally sending out important data such as gathered passwords and encryption keys.
3. A covert network might be *stealthy* in terms of preventing human detection at high volume levels by solely utilizing the ultrasonic frequency range.
4. A covert network might also be *stealthy* by steganographically hiding the transmission in the background noise (e.g. as already proposed for underwater networks [LvW08]). However, this might not be practical in air-based networks, where much smaller distances are overcome in comparison to underwater networks. Because of this, the user might be able to locate and, therefore, detect the transmissions (e.g. ventilation fan noise that is apparently produced by the speakers).

As the presented approach for a covert network does not feature steganography, computer-aided identification of the communications is feasible as shown in the following section.

3 Considerations on the detection and localization of covert audio transmissions

3.1 Audio profile of an unmodified laptop computer

Covert acoustical communications can be visualized with the help of a spectrum analyzer that produces a pattern similar to the pattern shown in Fig. 2. However, detection of covert acoustical communications from looking at a spectrogram can be a non-trivial task as other audio sources within the computing system are present that might be erroneously interpreted as a covert audio transmission. This fact is presented in Fig 5, where the audio profile of a laptop (type: Lenovo X201) is analyzed (without any covert communications).

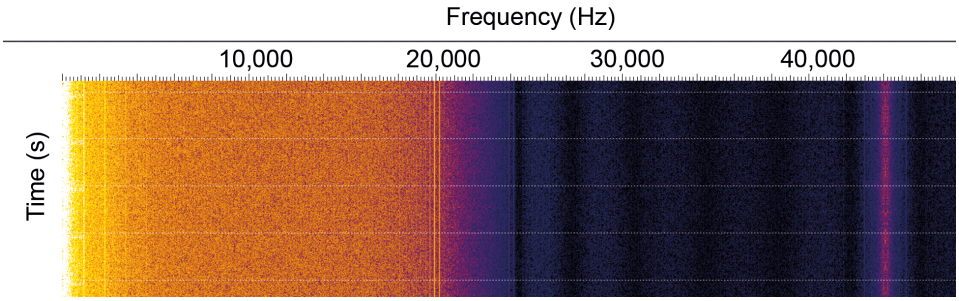


Figure 5: Spectrogram of sounds emitted by an unmodified laptop

The spectrogram was recorded with an external microphone (AKG C 1000 S) at a distance of 20 cm between the microphone and the laptops's ventilation opening. Although the microphone is primarily designed for recordings $\leq$ 20,000 Hz, it is also capable of recording some parts of the ultrasonic frequency range according to the manufacturer's specification [AKG]. The analysis reveals the presence of acoustical emanations that form a narrow band around 20,000 Hz and are apparently generated by the laptop's ventilation fan. Thus, the presence of ultrasonic emanations from unmodified laptops has to be considered in visual analyses. As LeMay and Tan [LT06] supposed, there might be further interesting emanations in the ultrasonic frequency range. However, as the sound processor of common laptops (i.e. Lenovo T400 and T410 types) gradually attenuates signals between 20,000 and 25,000 Hz [HG13], communication with the internal speakers and microphones can only occur in the frequency range below 25,000 Hz.

3.2 Experiment on detection and localization of covert audio transmissions

As an early warning system against covert acoustical communications, a frequency converter can be utilized to pitch down inaudible frequencies and produce an audible signal with each transmission. In a combined experiment on visual and acoustical detection of covert acoustical communications, we utilize the Rohde & Schwarz PR100 portable re-

ceiver that was originally designed for on-site radio monitoring [Roh]. Over a specially crafted connector, the PR100 is electrically connected with a microphone as its input device and configured to capture the frequency range around 20,000 Hz. A laptop is configured to output covert audio transmissions. The internal display of the PR100 is used for the visual detection mechanism (spectrum analyzer) and the internal speakers of the PR100 are used as output devices, while converting the recorded audio input back to humanly audible frequencies around 2,000 Hz. With comparable devices, detection of covert acoustical communications can be implemented as a mobile or stationary service. The same approach might also be implemented in software. The described detection and localization experiment is depicted in Fig. 6.

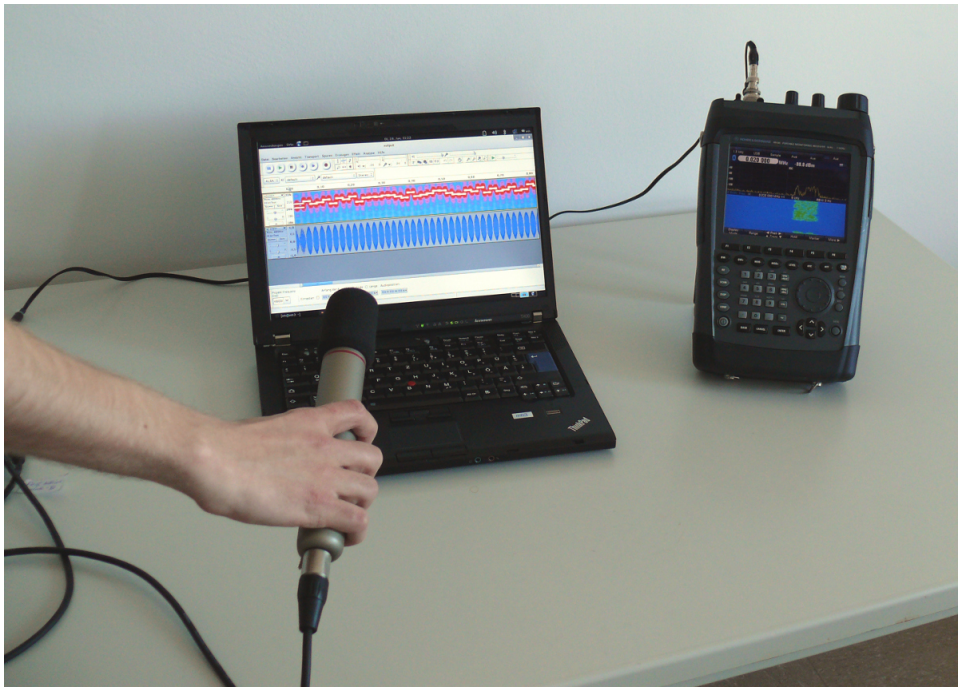


Figure 6: Locating the source of a covert audio signal with a mobile device (PR100)

As a result of the detection and localization experiment, the acoustical detection mechanism produces a specific audio output that is characteristic to the utilized frequency hopping sequence used in our experiments and, therefore, offers an easy-to-implement solution to detect this specific type of audio signal. The audio source could be exactly located by pointing the microphone in different directions and following the path of the loudest audio output. However, the acoustical detection mechanisms might also suffer from previously inaudible audio sources that could erroneously appear as an audio signal. Therefore, these detection mechanisms are most applicable if the characteristics (i.e. the signature, as explained in the following section) of the audio signal are previously known.

Moreover, if a highly advanced type of malware would hide acoustical communications in

a low signal-to-noise ratio (e.g. as shown in underwater networks [LvW08, vWLS+09]), the detection mechanisms presented so far might not be successful. In difference to underwater networks, we suppose this technique only to be feasible for ultrasonic sources of noise, as a speaker emitting background noise (e.g. the audible parts of the typical fan noise) could immediately attract the user's attention.

Finally, data transmissions might also be steganographically hidden within audible sound playback [PK00], but this would limit communications over the covert network to situations where audio playback is actually initiated by the user.

Detection of these types of audio signals can also be automated for use in an audio IDS (intrusion detection system) as described in the following section.

4 Implementation of an adaptive audio intrusion detection system

Based on a previously patented signal detection and autocorrelation method for radio communications [KLP12], we describe the implementation of an adaptive audio IDS. The underlying framework operates along the following principles:

1. A time slice of a prerecorded signal is extracted and analyzed.
2. Different signal characteristics are extracted from the signal for further analysis.
3. The captured signal characteristics are stored as a signature of a known signal.
4. During the operation of the audio IDS, live captured signal data is matched against the stored signatures.
5. Upon a match, an intrusion warning regarding the detection of a known audio signal is generated.

In our demonstration, an extracted time slice of approximately 2.5 seconds is selected from a covert audio transmission (see Fig. 7).

For automatic intrusion detection of the presented signal, the frequency shift utilized in the previously depicted frequency hopping sequence was found to be a suitable characteristic for automatic signal detection. Fig. 8 shows the results of the frequency shift analysis.

As can be seen in Fig. 8 the frequency shift analysis reveals a peak at 45.8 Hz, which matches the most prevalent frequency shift between two modulated audio tones. Moreover, local peaks at 91.6 Hz and other multiples of 45.8 are shown in Fig. 8. The observed frequency shift thus offers a reliable characteristic for the automatic detection of the presented audio signal and for related audio signals.

Alongside the frequency shift analysis, different methods and characteristics might be utilized for signature generation and matching (e.g. CA-CFAR, see also [Gie12]), as well as periodicity and structure repetition [Kur13]).

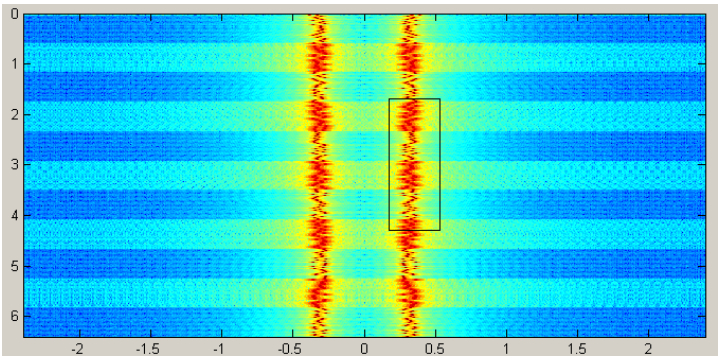


Figure 7: Time slice extracted from the recorded audio input

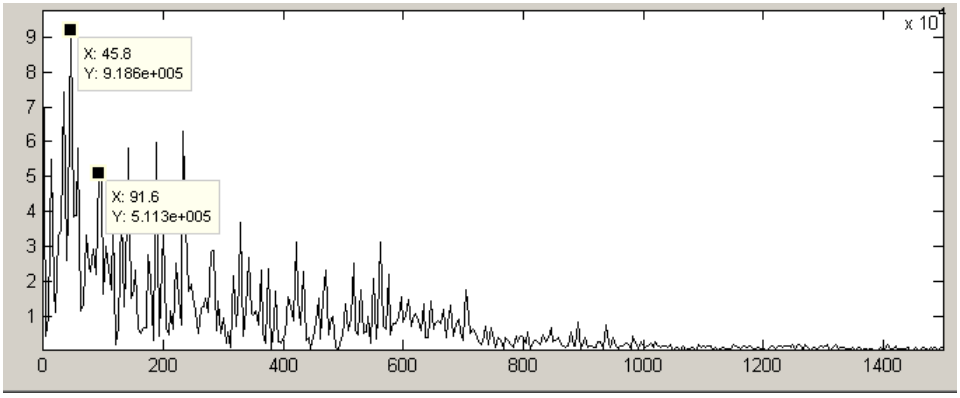


Figure 8: Automatically detecting covert audio signals by frequency shift analysis

The audio IDS is designed as an *adaptive* framework, as various signal characteristics can be chosen as detection patterns and new types of audio signals can easily be integrated into the framework. As we previously suggested, the audio IDS might be implemented within a trusted operating system component (i.e. as an audio intrusion detection guard [HG13]).

5 Related work

Acoustical networking has been suggested by several authors [MSTS05, NCPV13] and preliminary studies on ultrasonic communication in a very different context have been conducted [TOD+10, HSH+13]. Moreover, acoustical communication is commonly used in underwater networks [OAC+12]. In difference to these authors, we put covert acoustical communications and acoustical mesh networks (e.g. as present in underwater networks) into the context of operating system security [HK13b, HK13a], and covert networks [HG13], respectively. Additionally, we present the current state of covert networks.

6 Conclusion

We have discussed recent developments in air-based covert acoustical communications.

An alternative signal for the previously described ACS modem has been presented that operates solely in the ultrasound frequency range, although the maximum transmission range was found to decrease with common laptops. Still, the impact of the covert network might be greatly extended by the regular movements of these mobile devices. A new bandpass filter was presented with the intention to filter out potentially audible acoustical artifacts. The stealthiness of the original signal has been improved, but steganographic methods may additionally be applied to also hide the signal from visual and acoustical detection mechanisms as presented here.

Various techniques for the detection and localization of covert acoustical communication have been discussed and the implementation of an adaptive audio intrusion detection system has been presented. It was demonstrated that automatic detection of covert acoustical communications is feasible by comparing the characteristics of the live recorded emanations with the previously stored signatures of audio signals.

While covert networks might also be implemented over malicious hardware, we conclude that the software-based approach is much more feasible, as infection can occur over traditional means (e.g. over removable media, and over the internet, if the computer is temporary connected to the internet), and does not rely on tampering the supply chain for the purchase of new computers.

In summary, covert networks might arise as an upcoming trend in future malware and have to be considered in the design and implementation of future security critical computing systems.

Acknowledgment

We would like to thank Jörg Keller for helpful comments and Ivor Nissen for making this research possible. Special thanks goes to Frank Kurth for helpful information on signal detection methods. We further acknowledge Hans-Peter Stuch and Christian Schlich for helpful assistance regarding the detection and localization experiment.

References

[AKG] AKG. AKG C 1000 S Condenser Microphone. http://www.fullcompass.com/common/files/2186-akg_c1000s_specs.pdf, Accessed: 2013-12-01.

[Com13] Communications of the ACM. Experimental Malware Uses Inaudible Sound to Defeat Network Air Gaps. http://cacm.acm.org/news/170320-experimental-malware-uses-inaudible-sound-to-defeat-network-air-gaps/fulltext, Accessed: 2013-12-05, December 2013.

[Flo05] Elia Florio. When Malware Meets Rootkits. *Virus Bulletin*, 2005(12), December 2005.

[Ger04] Gergely Erdélyi. Hide'n'Seek? Anatomy of Stealth Malware. http://download.adamas.ai/dlbase/ebooks/VX_related/Hide%27n%27Seek%20Anatomy%20of%20Stealth%20Malware.pdf, Accessed: 2013-12-01, March 2004.

[Gie12] Ryan Thomas Gieleghem. Robust Acoustic Signal Detection and Synchronization in a Time Varying Ocean Environment. Master's thesis, Massachusetts institute of Technology, October 2012.

[GN12] Michael Goetz and Ivor Nissen. GUWMANET - Multicast Routing in Underwater Acoustic Networks. In *Military Communications and Information Systems Conference*, MCC '12, pages 1–8. IEEE, October 2012.

[Goo13] Dan Goodin. Scientist-developed malware prototype covertly jumps air gaps using inaudible sound. http://arstechnica.com/security/2013/12/scientist-developed-malware-covertly-jumps-air-gaps-using-inaudible-sound/, Accessed: 2013-12-02, December 2013.

[HG13] Michael Hanspach and Michael Goetz. On Covert Acoustical Mesh Networks in Air. *Journal of Communications*, 8(11):758–767, November 2013.

[HK13a] Michael Hanspach and Jörg Keller. A Taxonomy for Attack Patterns on Information Flows in Component-Based Operating Systems. In *Proceedings of the 7th Layered Assurance Workshop*, New Orleans, LA, USA, December 2013.

[HK13b] Michael Hanspach and Jörg Keller. In Guards we trust: Security and Privacy in Operating Systems revisited. In *Proceedings of the 5th ASE/IEEE International Conference on Information Privacy, Security, Risk and Trust*, PASSAT '13, Washington, DC, USA, September 2013. IEEE.

[HSH+13] Ragib Hasan, Nitesh Saxena, Tzipora Haleviz, Shams Zawoad, and Dustin Rinehart. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In *Proc. 8th ACM SIGSAC symposium on Information, computer and communications security*, ASIA CCS '13, pages 469–480, New York, NY, USA, 2013. ACM.

[IET07] IETF Network Working Group. RFC 4949. Internet Security Glossary, Version 2. http://tools.ietf.org/html/rfc4949, Accessed: 2013-12-16, August 2007.

[KLP12] Frank Kurth, Hans Günter Lehn, and Rolf Parting. Patent DE102009035524 B4. Verfahren zur Erkennung eines oder mehrerer Nutzsignale innerhalb eines Quellsignals (German language content). http://www.google.com/patents/DE102009035524B4?cl=de, Accessed: 2013-01-29, November 2012.

[Kur13] Frank Kurth. The shift-ACF: Detecting multiply repeated signal components. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, October 2013.

[Lam73] Butler W. Lampson. A note on the confinement problem. *Commun. ACM*, 16(10):613–615, October 1973.

[LT06] Michael D. LeMay and Jack Tan. Acoustic Surveillance of Physically Unmodified PCs. In *Proceedings of the 2006 International Conference on Security & Management*, June 2006.

[LvW08] G. Leus and P. van Walree. Multiband OFDM for Covert Acoustic Communications. *IEEE J.Sel. A. Commun.*, 26(9):1662–1673, December 2008.

[MSTS05] Anil Madhavapeddy, David Scott, Alastair Tse, and Richard Sharp. Audio Networking: The Forgotten Wireless Technology. *IEEE Pervasive Computing*, 4(3):55–60, July 2005.

[MTZ10] K. McCoy, B. Tomasi, and G. Zappa. JANUS: the genesis, propagation and use of an underwater standard. In *European Conference on Underwater Acoustics*, July 2010.

[NCPV13] Rajalakshmi Nandakumar, Krishna Kant Chintalapudi, Venkat Padmanabhan, and Ramarathnam Venkatesan. Dhwani: secure peer-to-peer acoustic NFC. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, SIGCOMM '13, pages 63–74, New York, NY, USA, August 2013. ACM.

[NG12] Ivor Nissen and Michael Goetz. Generic UnderWater Application Language (GUWAL) - Specification of Tactical Instant Messaging in Underwater Networks. Technical Report WTD71 - 0070/2012 FR, Research Department for Underwater Acoustics and Marine Geophysics, Kiel, Germany, December 2012.

[OAC+12] R. Otnes, A. Asterjadhi, P. Casari, M. Goetz, T. Husøy, I. Nissen, K. Rimstad, P. van Walree, and M. Zorzi. *Underwater Acoustic Networking Techniques*. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012.

[ON10] Ramanagouda Odugoudar and Ivor Nissen. Ad-hoc Network Emulation Framework for Underwater Communication Applications. In *Proc. 5th ACM International Conference on Underwater Networks & Systems*, Woods Hole, MA, September 2010. ACM.

[Owa13] Nancy Owano. Authors explore security threat of covert acoustical mesh networks in air. `http://phys.org/news/2013-12-authors-explore-threat-covert-acoustical.html`, Accessed: 2013-12-02, December 2013.

[PK00] Natalie Packham and Frank Kurth. Transport of context-based information in digital audio data. In *Audio Engineering Society 109th Convention*, September 2000.

[Roh] Rohde & Schwarz. R&S PR100 Portable Receiver On-site radiomonitoring from 9 kHz to 7.5 GHz. `http://www.rohde-schwarz.de/file/PR100_bro_en.pdf`, Accessed: 2013-12-02.

[TOD+10] David Tofsted, Sean O'Brien, Sean D'Arcy, Edward Creegan, and Scott Elliott. An Examination of the Feasibility of Ultrasonic Communications Links. Technical Report ARL-TR-5200, Army Research Laboratory, White Sands Missile Range, NM, USA, June 2010.

[vWLS+09] Paul van Walree, Thorsten Ludwig, Connie Solberg, Erland Sangfelt, Arto Laine, Giacomo Bertolotto, and Anders Ishøy. UUV Covert Acoustic Communications. In *Proceedings of the 3rd conference on Underwater Acoustic Measurements: Technologies and Results*, June 2009.

Consumer Participation in Online Contracts – Exploring Cross-Out Clauses

Sebastian Luhn[a], Ina Bruns[b], Rainer Böhme[a]

[a]IT Security Research Group, Department of Information Systems
[b]Institut für Informations-, Telekommunikations- und Medienrecht
Westfälische Wilhelms-Universität Münster, Germany

{sebastian.luhn, ina.bruns, rainer.boehme}@uni-muenster.de

Abstract: E-commerce is reality. Millions of consumers buy goods or subscribe to services online. However, they are often presented with take-it-or-leave-it decisions: consumers must accept standard business terms and privacy policies as they are or the contract cannot be concluded. We explore the idea of letting the consumer cross out unwanted clauses online, giving back what was possible for offline contracts. We built a prototype and conducted a user study of 24 face-to-face tests with subsequent interviews in public space, applying both a between-subject and a within-subject experimental design. Results show that the participants of the study appreciated the idea and actively made use of it. Additionally, we observe a tendency that users read contracts more thoroughly if they know that they can alter them. This may help coming closer to the intended notion of informed consent when contracting online.

1 Introduction

Concluding contracts on the Internet is a well-established way for consumers to buy goods and subscribe to services. In 2013, 61 % of consumers in the EU bought online at least once [SR13]. A problem with this approach is that consumers often face a take-it-or-leave-it decision: they can accept the terms of the contract as a whole or choose to not use the service. German law, however, generally allows consumers to cross out clauses they want to exclude from the contract (cf. [BL13] for legal aspects of cross-out clauses online), giving the consumer a basis for negotiation of contract terms with the service provider. This is easily done on paper, but most online contract forms do not offer this option. We explore ways to give it back.

In this paper, we report our experience with a prototype that allows users to cross out certain clauses pre-defined by the service provider. Furthermore, we present a user study that evaluates our prototype. This study seeks to answer three research questions. First, behavioural aspects are examined by observing how the study participants interact with our prototype. Second, we analyse self-reported attitudes towards the general idea of crossing out clauses. Third, concrete design options are evaluated by pairwise comparisons between variants of the prototype.

Our research applies to contract clauses of various kinds, including Standard Business Terms (SBT), End User Licence Agreements (EULA) and privacy policies. We refer to them as *contract terms* and differentiate only where the specific kind of terms matters.

Online contracts also suffer from the problem that people tend to read their terms only superficially or not at all [JP04, MC04, GG07]. To address this problem, it has been proposed to shorten contracts or provide non-legally binding versions that are easier to understand [KCBC10, GG07]. In this paper, we examine whether our prototype can help solving this problem by encouraging users to interact with contracts. We present evidence that, indeed, people tend to read the contract more thoroughly if they can alter some of its clauses.

The rest of this paper is structured as follows. Section 2 reviews related research. We present our prototype in Section 3. The user study is described in Section 4. Section 5 discusses the results and concludes the paper.

2 Related Work

There is an economic argument to be made regarding the improvement of contract terms. McDonald and Cranor estimate the theoretical opportunity cost for the time to read privacy policies under the assumption that all consumers read them. They add up to a stunning US$ 780 billion per year for US consumers, about 40 times the value of the online advertising market [MC08]. User interaction with contract terms presented online have been subject to research for quite a while. We broadly distinguish prior work in approaches *reducing* user involvement and *improving* user involvement.

2.1 Reducing User Involvement

A reason for the need to reduce user involvement is given by Böhme and Grossklags. They argue that the user's attention is a scarce resource which is best spent on a few important instead of many, partly trivial decisions. Otherwise, important decisions could suffer [BG11].

One way of reducing the number of decisions is by automating them. Formal languages can be used for this task, some of which are presented in the following. All of them deal with privacy policies. P3P, shorthand for Platform for Privacy Preferences, is a protocol allowing websites to state their privacy policy in a standardised formal language [CW07]. Additionally, it allows users to state their preferred policy. When a user visits a website, its privacy policy is compared to the user's preferences and differences shown to the user.

The EU-funded EnCoRe project aims to strengthen the user's control over her personal data processed on websites with a formal language and a supporting infrastructure on the service provider's side exceeding P3P's server-side mechanisms. In [PCG09], the authors describe the foundations of the project. Similar to an earlier approach in [Hom05],

XACML is used to specify privacy policies and to enforce purpose binding of personal data. Additionally, [ACGP10] describes revocation mechanisms to prevent further usage of personal data.

Another privacy policy language is the Enterprise Privacy Authorization Language (EPAL), developed by IBM in 2003. It uses XACML to define conditions for data usage. Ashley et al., some of its authors, view EPAL as being complementary to P3P [AHK+03].

The presented approaches accomplish the task of reducing user involvement. However, reaching widespread adoption can be challenging. For example, a survey on P3P support in 2006 showed that only about 10% of the top-20 websites found by searching for one of about 20,000 popular search terms have P3P policies [ECC06]. Without a critical mass supporting these languages, there is a chicken-and-egg problem: browser vendors are reluctant to implement features that websites do not use, and vice versa. Additionally, users without deep technical knowledge are often alienated by the complexity of policy languages [Cen00].

2.2 Improving User Involvement

Concluding a contract should follow the concept of *informed consent*. In many situations, consent is required to be explicit, voluntary and specific. Explicit means that consumers give consent by a distinct action. Voluntary means consumers do so out of free will. Specific means that consent is given to a well-defined usage scenario (e.g., [FLM05]). The concept also has a policy dimension as its role is about to be strengthened in the EU's data protection reform [Com12]. Kim suggests coming closer to the notion of informed consent could be achieved by having the user confirm every usage of their data one-by-one when first dealing with a service provider. This would imply that more invasive privacy policies need more confirmations and thus incentivises service providers to implement privacy-friendlier policies [Kim10].

Several studies on the usability of contract texts show that informed consent is often absent in practice. The authors of [JP04, MC04, GG07] all conclude that real-world contract texts are often unreadable for their length and complex language. Studies on the behaviour of users dealing with contract texts show that they are only superficially read, if at all [GG07, MC04]. When asked, consumers also claim that the lack of choice when wanting to use a product or service causes them to not read contract texts [PB11]. Giving the user a choice in parts of the contract leads to different behaviour depending on whether options are presented as opt-in or opt-out. Empirical studies, e.g. by Johnson et al., have shown that default options have a measurable effect on the option chosen by users [JBL02].

Even without changing anything in the contract texts themselves, it is possible to get users to read them more thoroughly. Plaut and Bartlett manipulated the beliefs users held in contract texts, e.g., by telling them beforehand they had a choice on its content or that it was different from common contract texts while still using the same text. Users primed with this information read the texts for a longer time. On the opposite, telling them the service provider was reputable decreased the reading time [PB11].

Approaches to improve the usability of contract texts differ by the type of contract. For privacy policies, there exist concise representations showing the type of data being processed and the purpose. For example, Petterson et al. developed a *Send Data dialogue* showing this information as well as retention options and a link to a verbose privacy policy [PFHD+05]. Kelley et al. experimented with representing the information in table form, additionally colouring the table cells according to the level of data use. In a study testing how much information was remembered from reading privacy policies, users who dealt with the table version could answer more questions correctly than those who read common verbose privacy policies [KCBC10].

For EULAs, Kay and Terry propose that augmenting the text is better than substituting it by a different representation or a shorter version of the text. They augment contract texts by adding layers, i.e. collapsible parts, colours, and pictographs to emphasise certain parts of the text. In a user study, they show that the augmented version of the original text is not only read longer than a control group contract text, but also longer than a shortened summary they provided as an alternative [KT].

All of the studies presented are meant to give an overview, not the complete picture. However, to the best of our knowledge, we are not aware of any studies that come close to our approach of crossing out clauses like on a paper contract.

3 The Prototype

In this section, we present a prototype that allows crossing out clauses. The idea behind our approach is to be evolutionary rather than revolutionary—its user experience is known from offline contracts.

3.1 Implementation

The prototype is designed after typical mobile phone service provider websites, which was tested and confirmed in the user study. A screenshot of the main page is provided in Appendix C. On the website, users can enter their credentials. The contract terms are placed below. Optional clauses can be crossed out in this area. Each optional clause is inside a HTML `span` environment.

Crossing out clauses is handled in JavaScript. A clause is selected by clicking on or marking at least part of it. If mandatory clauses are marked, nothing happens. A clause is either crossed out directly after it is marked or by clicking on a button (see below). Additionally, its `span` environment's `class` name is changed. After crossing out clauses, the user clicks on a *Check Data* button. The crossed-out clauses are sent to the server hosting the website via JSON. Further steps could be taken at this stage, e.g., the altered contract could be evaluated by the service provider. This feature is not implemented in the prototype. Afterwards, the clauses are sent back to the client and confirmed by the user.

3.2 Content and Variants

We created the SBT under legal consideration (cf. [BL13]) to look like common SBT of mobile phone companies. However, we formulated easy-to-understand SBT to avoid users giving up on reading the text due to not understanding it (cf. Section 2.2). Two clauses could be crossed out: paying per direct debit and the automatic renewal of the contract. Additionally, we made some options available as choices outside the text, e.g., the duration of the contract and its cancellation period.

We designed the privacy policy to be concise with three main options. Users could choose whether they wanted to get advertisements from the mobile phone service provider or from third parties. Additionally, they could choose to not be contacted for market research purposes. For each of the options, four different contact ways were available: e-mail, standard mail, text message, and phone calls.

We developed several variants of the prototype. It either had one, two or no buttons to cross out clauses. For the variants with one or two buttons, clauses were crossed out by clicking on a button. The same button undid crossings in the one-button variant. Clauses got crossed out as soon as they were marked in the variants without buttons.

There either was a checkbox to emphasise optional clauses or they were always emphasised. Finally, users could either only cross out parts of the privacy policy clauses (i.e., only contact ways) or whole clauses. These variants were created because the latter is more natural like pen & paper while the former is more practical and automatisable. We also tested a control group that could choose some of the optional clauses via checkboxes.

4 User Study

4.1 Goals

The goals of the user study are defined by three research questions (RQs):

- RQ 1: How do consumers deal with contract texts in order forms?
- RQ 2: How do consumers evaluate the option to cross out clauses?
- RQ 3: How do consumers evaluate different design options?

For RQ 1, we observed the behaviour of the participants of the study during the simulated contract formation. Thus, we know how thoroughly the contract terms were read. We can then compare the results to the literature to see whether the possibility to interact with the contract encourages users to read it in more detail.

For RQ 2, we asked whether the participants liked the idea of crossing out unwanted clauses and if they would use it. We compared this with their behaviour during the simulated contract formation. Possibly, contradicting results could tell something about the difference between liking the idea in theory and using our prototype.

Crossing out clauses can be implemented in various ways. For RQ 3, we tested different variants of our prototype described in detail in Section 3.2 (cf. Table 2 in Appendix B for information on variant distribution in the study). We stated some expected results beforehand: buttons help users understand they are able to cross out clauses. Regarding the emphasis of clauses we expect two effects. On the one hand, clauses that are already emphasised should be more visible. On the other hand, having to tick a box possibly serves as a hint for the user. Being only able to cross out parts of the privacy policy clauses could lead to fewer crossed out clauses or a worse evaluation of the prototype.

4.2 Design of the Qualitative User Study

Our study consisted of 24 participants, of whom 17 were male and seven female. Eleven were recruited from a law lecture, the others by approaching them spontaneously. Apart from students, we also asked secretaries and librarians to broaden our sample spectrum. For more details on the composition of the sample, please refer to Table 1 in Appendix B. Each participant performed the study individually, i.e. there was exactly one participant sitting face-to-face to a conductor, using the prototype on a laptop. RQ 3 was answered both by applying a between-subject and a within-subject design. Most participants only used one variant of the prototype but we let some compare two variants directly (see below).

The study was divided into four parts: First, each participant answered some general questions regarding demographics, knowledge of computers and experience with mobile phone contracts. She was informed that she should act as if she would sign a real mobile phone contract. Then, she performed the simulated contract formation. We monitored her choices and how long and thoroughly she read different parts of the contract website to answer RQ 1. Afterwards, we interviewed her on the contract formation, answering mainly RQs 1 and 2. Lastly, we let her fill out a questionnaire to evaluate the prototype itself, i.e., the implementation, which concerned RQs 2 and 3.

Interview. In the interview, we first asked about problems during the contract formation and what the participants thought the intention of the prototype was. We then checked facts from the contract texts to determine how thoroughly the participants had read them, a part of RQ 1. After that, we asked questions regarding RQ 2 as mentioned in Section 4.1. We also asked whether they thought this technique was easy enough to work with, would lead to reading contract texts more thoroughly and how people became aware of what they could do. Lastly, we asked eight people to perform a consecutive test and compare another variant of the prototype to the one they had just tested, answering RQ 3.

Questionnaire. We used an adapted (cf. Appendix A) version of [MRT13a], a questionnaire using the Components of User Experience (CUE) model to measure the quality of interactive products. The questionnaire is presented in detail in [MR13] and was validated in [MRT13b] and [MRT13c]. All of its items belong to one of five categories: The *usefulness* of a product describes whether users can achieve a certain goal with it. The question answered by this measure is if *what* the product does is good. The *usability* of a product describes how efficiently the user can reach her goals. The question answered by this measure

is whether *how* the product implements its functionality is done well. *Aesthetics* measures whether its design looks attractive to the user. Here, functionality is of no concern. *Negative emotions* can be split up into *passive* and *active* negative emotions. Passive emotions are, e.g., getting tired by the product, while an active negative emotion could be being frustrated by it. Lastly, *loyalty* is a measure to determine whether the product encourages the user to be used again.

4.3 Results

The results of the user study are presented along the research questions stated in Section 4.1.

Behaviour of test subjects (RQ 1). During our study, most participants read at least the privacy policy in great detail. The SBT were often read in lesser detail, albeit most participants still remembered facts from the SBT correctly. One participant probably even read the SBT of a mobile phone contract for the first time, as she wondered why a request to a credit agency was included, a standard practice in mobile phone contracts.

On the other hand, six participants did not read the contract terms at all or only very superficially, even though they were primed with testing a consumer-friendly version of a mobile phone website. This behaviour is reflected in the literature, e.g. in [JP04], [MC04], and [GG07], but still unexpected considering the circumstances of the study. Still, the number of participants showing this behaviour is relatively small. One of the problems mentioned was that the contract terms were only available by clicking on a link instead of being directly visible. Additionally, participants claimed they were not being informed about their possibilities even though information about crossing out clauses was provided directly above the contract terms. Only three participants claimed this information made them aware of what they could do. Additionally, participants claimed the contract terms were inside containers that were too small. Optional clauses should be more emphasised and SBT only acceptable after at least scrolling through them.

All but two participants who noticed they could cross out clauses did so at least in the privacy policy, all in all 18 out of 20. Four out of the six who only superficially read the contract terms never noticed they were able to cross out clauses. In the SBT where there were far less options, only few participants crossed out clauses. Interestingly, one participant claimed he did not read clauses that came directly after crossable ones.

Crossing out clauses was problematic in six cases. Aside from variant-specific problems described below, marking clauses correctly was the biggest problem. Participants often asked how they could mark clauses even before trying for themselves. If they marked clauses in a way such that the prototype did nothing (cf. Section 3.1), all but one did not try again before being helped.

General attitudes (RQ 2). Generally, the idea of crossing out clauses was near unanimously seen positively. All but one participant would prefer having such an option, all but two would use it themselves. The participant favouring the idea without wanting to use it claimed that in reality he would not trust such a novel feature. Still, there were more participants who claimed they would cross out clauses than who actually did so. We asked

the participants who showed a contradictory behaviour for an explanation. Three people claimed they did not see how to cross out clauses, while the fourth person simply did not find anything in the contract texts he wanted to cross out.

However, while almost all participants in general liked the idea, there were mainly three critical remarks when asked whether crossing out clauses was simple enough. The first critical remark was that, especially in lengthy SBT, the few optional clauses can be hard to find even if they are emphasised. People have to scroll through the entire text to find all clauses. Thus, easily readable, short SBT were claimed to be more important than crossing out clauses by two participants.

The second critical remark was that having all optional clauses as checkboxes would be easier. On the other hand, all participants stating this explicitly said it was only true if all clauses were available as checkboxes, which is rather unrealistic considering the amount of optional clauses. Additionally, one of the goals of our idea is to make consumers aware of the contract terms. This can be hard with checkboxes due to users blindly checking them (cf. Section 2.2). Finally, all clauses in our contract were positive for the service provider when not crossed out. This was criticised as well. It has to be kept in mind, however, that the usual alternative to this would be to have no options at all.

Two answers dominated when we asked what the goal of the prototype was: offering a transparent privacy- and consumer-aware alternative to standard online contract formation and getting consumers to read contract terms more thoroughly. Telecommunication companies trying to gain trust they lacked in the eyes of the participants was also mentioned. One participant, however, saw our prototype cynically and claimed it only suggested to offer autonomy of decision where in reality there was none.

All but one participant claimed they would read contract texts in more detail given the option to cross out clauses. Most participants claimed it would also lead to a more thorough examination of contract texts by the general public, but only 16 gave an unconditionally positive answer to this question.

There was no specific part of the prototype that made participants aware of their possibilities. All of the possible hints, e.g., buttons, the checkbox to emphasise clauses, emphasised clauses themselves, the structure of the website, and the information text above the contract texts, were mentioned about evenly.

Additionally to asking about the general idea of crossing out clauses, we also wanted to test our implementation (cf. screenshots in Appendix C). One of the evaluation methods used was the questionnaire described in Section 4.2. An overview of the results can be found in Figure 1.

Overall it can be seen that the participants evaluated the prototype positively. In all categories the median is in the upper half of the respective scale. The usefulness of the prototype was evaluated especially positively. With a median of 6 and no values below 4, it was always rated at least neutral. The prototype's Usefulness is higher rated than usability, aesthetics and loyalty. This order is good, as usefulness is clearly the most important aspect of our study as it also assesses the idea. Weaker ratings of the usability and the aesthetics can be explained by a prototype being tested.

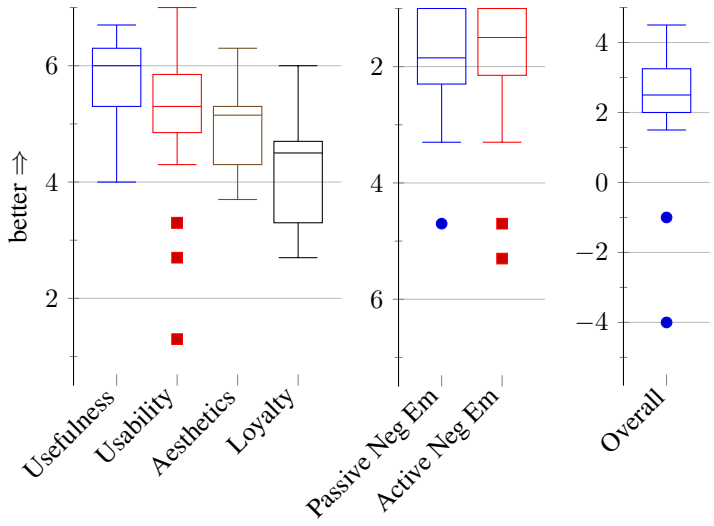


Figure 1: Results of the questionnaire. *Neg Em* are negative emotions, therefore lower values are better and the scale is inverted. $(n = 24)$

Loyalty is the least important category and rather hard to measure correctly due to the nature of the product being tested—people probably see a contract website as something they have to use, not something they would want to use regularly.

One of our goals was to reach a low level of negative emotions, in which we succeeded. Except for one and two outliers, respectively, both aspects of negative emotions were rated on a low level. The same is true for the overall evaluation which is positive except for two outliers.

Variants of the prototype (RQ 3). Contrary to our expected result, variants without buttons were preferred to variants with buttons. The main reason stated for this was that without buttons, one could experiment with marking the clauses and immediately see a result instead of having to click on a button first. Only two participants preferred the variants with buttons because it served as a hint for what they could do.

Having to check a checkbox to emphasise clauses that can be crossed out was generally preferred to clauses being emphasised from the beginning due to the checkbox serving as a hint. We expected this, but it nevertheless contradicts the result on ease-to-use vs. explicating functionality we generally saw for buttons. This is probably due to checking a checkbox once is easier than having to click on a button for each clause that is crossed out.

Versions where it was possible to cross out whole clauses in the privacy policy were not evaluated differently than those where only parts of them could be crossed out. Additionally, in the questionnaire, differences of the scores between variants were very small. This could possibly be explained by the relatively small number of participants or the fact that while there were differences in preference regarding the variants, they were considered relatively

minor.

The consecutive tests (cf. Section 4.2) performed with eight of the participants mainly explicated results that were implicitly stated in the interview.

5 Discussion and Conclusion

We have explored, to the best of our knowledge for the first time, the idea of allowing users to cross out clauses in standard business terms and privacy policies. This option enriches online contracts with part of the negotiation power consumers enjoy offline. A user study with a prototype implementation reveals that the participants like the idea of being able to cross out clauses and actively make use of it. When asked for their impression, most participants expect that this option leads people to deal with contract terms more thoroughly. Therefore, we believe that our idea helps to come closer to the intended notion of *informed consent* between contracting partners.

However, there are a few limitations attached to the generally positive results. As also mentioned by participants of our study, the prototype does little to enforce simpler or shorter SBT and privacy policies. We used shorter than average texts for our study, but the question remains whether our results generalise to longer, more complex SBT with fewer options. Furthermore, to implement our system in an existing business architecture, this architecture must be adapted to accept and manage different SBT and privacy policies for each user. This aspect was deliberately excluded from our study. In practice, implementing the new functionality and handling diverse contacts with more consumer-friendly terms imposes additional costs on the service provider. Unless these costs are offset by competitive advantages, the prospects for voluntary adoption by the industry are dim and additional incentives might be needed. Finally, part of the success of our prototype could be caused by its novelty. Over time, a habituation effect may emerge that attenuates the observed benefits.

To conclude, our study shows that getting the user to read contracts more thoroughly is possible by giving her the option to cross out unwanted clauses. By staying close to a familiar usage concept, users have few problems when interacting with our prototype. Future research could focus on the implications for service providers and the factors affecting their decision to adopt.

Acknowledgements

This research has been funded by the Competence Centre for Consumer Research *(Kompetenzzentrum für Verbraucherforschung, KVF)* of North-Rhine Westphalia as part of a project jointly led by Rainer Böhme, IT Security Research Group, Department of Information Systems, University of Münster and Franziska Boehm, Institut für Informations-, Telekommunikations- und Medienrecht, University of Münster.

References

[ACGP10] I. Agrafiotis, S. Creese, M. Goldsmith, and N. Papanikolaou. Reaching for informed revocation: Shutting off the tap on personal data. In M. Bezzi, P. Duquenoy, S. Fischer-Hübner, M. Hansen, and G. Zhang, editors, *Privacy and Identity Management for Life*, pages 246–258. Springer, 2010.

[AHK+03] P. Ashley, S. Hada, G. Karjoth, C. Powers, and M. Schunter. The Enterprise Privacy Authorization Language (EPAL) – How to enforce privacy throughout an enterprise, 2003. Available at: http://www.w3.org/2003/p3p-ws/pp/ibm3.html. Last accessed: 2013-11-18.

[BG11] R. Böhme and J. Grossklags. The security cost of cheap user interaction. In *Proceedings of the New Security Paradigms Workshop*, pages 67–82, Marin County, CA, 2011. ACM.

[BL13] I. Bruns and S. Luhn. Verbraucherschutz durch Mitentscheidung bei Online-Verträgen. In J. Taeger, editor, *Law as a Service (LaaS) — Recht im Internet- und Cloud-Zeitalter, Tagungsband Herbstakademie 2013*, volume 2, pages 859–874, Berlin, 2013.

[Cen00] Electronic Privacy Information Center. Pretty Poor Privacy: An Assessment of P3P and Internet Privacy, 2000. Available at: http://epic.org/reports/prettypoorprivacy.html. Last accessed: 2013-11-18.

[Com12] European Commission. Proposal for a regulation of the European Parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation) (COM (2012) 11 final), 2012.

[CW07] L. F. Cranor and R. Wenning. P3P: The Platform for Privacy Preferences, 2007. Available at: http://www.w3.org/P3P/. Last accessed: 2013-11-18.

[ECC06] S. Egelman, L. F. Cranor, and A. Chowdhury. An analysis of P3P-enabled web sites among top-20 search results. In *Proceedings of the 8th International Conference on Electronic Commerce*, pages 197–207. ACM, 2006.

[Eur12] Eurostat. Computer skills in the EU27 in figures, 2012. http://epp.eurostat.ec.europa.eu/cache/ITY_PUBLIC/4-26032012-AP/EN/4-26032012-AP-EN.PDF.

[FLM05] B. Friedman, P. Lin, and J. K. Miller. Informed consent by design. In L. F. Cranor and S. Garfinkel, editors, *Security and Usability*, pages 495–521. O'Reilly, 2005.

[GG07] J. Grossklags and N. Good. Empirical studies on software notices to inform policy makers and usability designers. In S. Dietrich and R. Dhamija, editors, *Financial Cryptography and Data Security*, volume 4886 of *Lecture Notes in Computer Science*, pages 341–355. Springer, 2007.

[Hom05] W. Hommel. Using XACML for privacy control in SAML-based identity federations. In J. Dittmann, S. Katzenbeisser, and A. Uhl, editors, *Communications and Multimedia Security*, volume 3677 of *Lecture Notes in Computer Science*, pages 160–169. Springer, 2005.

[JBL02] E. J. Johnson, S. Bellman, and G. L. Lohse. Defaults, framing and privacy: Why opting in-opting out. *Marketing Letters*, 13(1):5–15, 2002.

[JP04] C. Jensen and C. Potts. Privacy policies as decision-making tools: an evaluation of online privacy notices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 471–478. ACM, 2004.

[KCBC10] P. G. Kelley, L. Cesca, J. Bresee, and L. F. Cranor. Standardizing privacy notices: an online study of the nutrition label approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1573–1582. ACM, 2010.

[Kim10] N. Kim. 'Wrap contracts and privacy. In M. Genesereth, R. Vogl, and M.-A. Williams, editors, *AAAI Spring Symposium on Intelligent Privacy Management*, pages 110–112, Menlo Park, CA, 2010.

[KT] M. Kay and M. Terry. Textured agreements: re-envisioning electronic consent. In *Proceedings of the Sixth Symposium on Usable Privacy and Security*, pages 13:1–13:13.

[MC04] G. R. Milne and M. J. Culnan. Strategies for reducing online privacy risks: Why consumers read (or don't read) online privacy notices. *Journal of Interactive Marketing*, 18(3):15–29, 2004.

[MC08] A. M. McDonald and L. F. Cranor. The cost of reading privacy policies. *I/S: A Journal of Law and Policy for the Information Society*, 4:543–565, 2008.

[MR13] M. Minge and L. Riedel. meCUE – Ein modularer Fragebogen zur Erfassung des Nutzungserlebens. *Mensch & Computer 2013: Interaktive Vielfalt*, pages 89–98, 2013.

[MRT13a] M. Minge, L. Riedel, and M. Thüring. meCUE, 2013. Available at: http://mecue.de/download.html. Last accessed: 2013-11-18.

[MRT13b] M. Minge, L. Riedel, and M. Thüring. Modulare Evaluation interaktiver Technik. Entwicklung und Validierung des meCUE Fragebogens zur Messung der User Experience. In E. Brandenburg, L. Doria, A. Gross, T. Güntzler, and H. Smieszek, editors, *Grundlagen und Anwendungen der Mensch-Technik-Interaktion. 10. Berliner Werkstatt Mensch-Maschine-Systeme*, pages 28–36. Universitätsverlag der TU Berlin, 2013.

[MRT13c] M. Minge, L. Riedel, and M. Thüring. Und ob du wirklich richtig stehst... Zur diskriminativen Validität des User Experience Fragebogens "meCUE". In S. Boll, S. Maaß, and R. Malaka, editors, *Mensch & Computer 2013 – Workshopband*, pages 137–144. Oldenbourg Verlag, 2013.

[PB11] V. C. Plaut and R. P. Bartlett III. Blind Consent? A social psychological investigation of non-readership of click-through agreements. *Law and Human Behavior*, 36(4):1–23, 2011.

[PCG09] N. Papanikolaou, S. Creese, and M. Goldsmith. Policy refinement checking. In *Ninth International Workshop on Automated Verification of Critical Systems (AVoCS 2009)*, pages 253–256. Swansea University, 2009.

[PFHD+05] J. S. Pettersson, S. Fischer-Hübner, N. Danielsson, J. Nilsson, M. Bergmann, S. Clauss, T. Kriegelstein, and H. Krasemann. Making PRIME usable. In *Proceedings of the 2005 Symposium on Usable Privacy and Security*, pages 53–64. ACM, 2005.

[SR13] H. Seybert and P. Reinecke. Internet use statistics – individuals, 2013. http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Internet_use_statistics_-_individuals.

A Questionnaire Details and Adjustments

The *meCUE* questionnaire [MRT13a] consists of statements regarding the product measured on a seven-stepped Likert scale. Additionally, an overall grade ranging from -5 to 5 can be given. We excluded all questions regarding *status* and *binding*, as contract websites are not prone to change the social status of a user or become an integral part of her life. Additionally, we left out questions on positive emotions. Our goal instead was to not invoke negative emotions, as we concluded this was the measure a contract website could achieve. Lastly, we excluded questions on intention of use, as they were all concerned with long-time repeated usage. This left us with questions on usefulness, usability, aesthetics, negative emotions and loyalty as well as an overall mark.

B Details on the Study Design

Dimension	Manifestation
Participants	24 overall 17 male 7 female
Profession of participants	11 from law lecture Secretaries, librarians, students, PhD candidates
Age	21–60
Computer experience	Above average, styled after Eurostat questionnaire [Eur12]. People who... copied files: 24 created presentations: 24 created Excel sheets: 20 implemented a computer program: 5
Mobile phone contract experience	22 have had mobile phone contracts 4 lastly contracted online 4 on telephone 16 in a brick-and-mortar store

Table 1: Stylized facts of the participants of the user study

Element	Variants
Buttons	7× 0 buttons 10× 1 button 7× 2 buttons
Clauses	8× only parts of privacy policy clauses 16× whole privacy policy clauses

Table 2: Variants of the prototype tested

C Prototype Screenshot

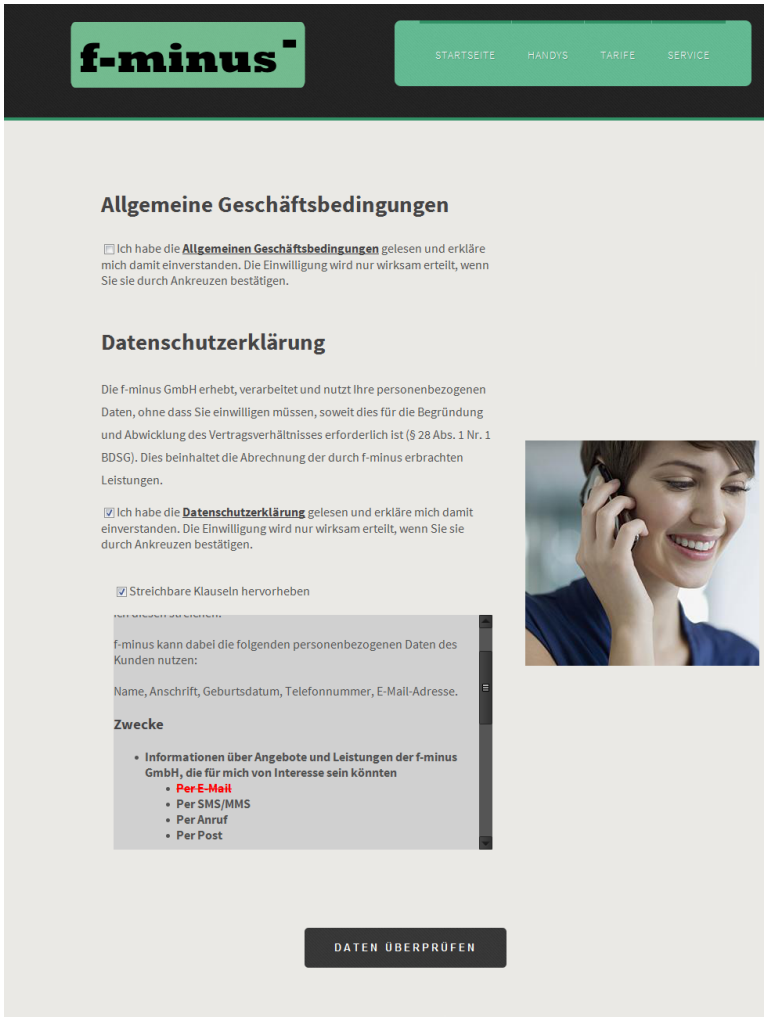


Figure 2: Screenshot of the prototype website. Variant with no buttons, option to emphasise clauses and whole clause crossable

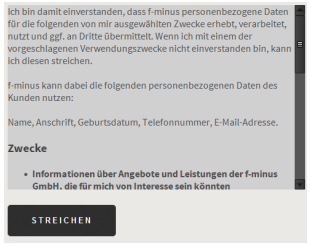
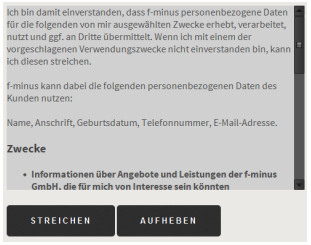
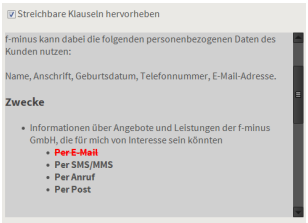
Element	Alternatives to Figure 2
Buttons	 One button  Two buttons
Clause marking	 Only parts of each privacy policy clause (contact way) can be struck, not the whole clause
Emphasis	 Clauses automatically emphasised without checkbox

Table 3: Screenshots of alternative variants

Verhaltensanalyse zur Erkennung von Missbrauch mobiler Geldtransferdienste*

Roland Rieke[1,2], Maria Zhdanova[2], Jürgen Repp[2], Romain Giot[3,4], Chrystel Gaber[5]

[1]Philipps-Universität Marburg, Marburg, Germany
[2]Fraunhofer SIT, Darmstadt, Germany
[3]Université Bordeaux, LaBRI, UMR 5800, F-33400 Talence, France
[4]CNRS, LaBRI, UMR 5800, F-33400 Talence, France
[5]Orange, Caen, France
{roland.rieke, maria.zhdanova, juergen.repp}@sit.fraunhofer.de,
romain.giot@u-bordeaux1.fr, chrystel.gaber@orange.com

Abstract: Die fortlaufende Überwachung von Transaktionen auf Geldwäscheverdacht ist Finanzinstituten in Deutschland und anderen Ländern vorgeschrieben. Smurfing ist eine Form der Geldwäsche, bei der durch den Transfer vieler kleiner Geldbeträge auf unterschiedlichen Wegen mit der Hilfe von Strohmännern ein hoher Geldbetrag unauffällig transferiert werden soll. In dieser Arbeit betrachten wir das Smurfing-Risiko im Rahmen mobiler Geldtransferdienste. Insbesondere beschreiben wir eine Methode zur vorbeugenden Sicherheitsanalyse zur Laufzeit, welche das Prozessverhalten in einem Geldtransfer-Service in Bezug auf Transaktionen beobachtet und versucht, es mit dem erwarteten Verhalten zu vergleichen, welches durch ein Prozessmodell vorgegeben ist. Wir analysieren Abweichungen von der vorgegebenen Verhaltensspezifikation auf Anomalien, die einen möglichen Missbrauch des Finanzdienstes durch Geldwäscheaktivitäten anzeigen. Wir bewerten die Anwendbarkeit der Vorgehensweise und beschreiben Messungen der Rechen- und Erkennungsleistung eines prototypischen Werkzeugs basierend auf realen und simulierten Betriebsprotokollen. Das Ziel der Experimente ist es, basierend auf Eigenschaften des realen Finanzdienstes, Missbrauchsmuster in synthetisiertem Prozessverhalten mit eingefügten Geldwäscheaktivitäten zu erkennen.

1 Einleitung

Dienste im Bereich mobiler Geldtransfer (MGT) sind ein wachsendes Marktsegment, insbesondere in Entwicklungsländern, in denen das Netz der Bankfilialen nicht so dicht ist. So hatte beispielsweise M-Pesa, welches 2007 in Kenia eingeführt wurde, im Dezember 2011 bereits etwa 19 Millionen Teilnehmer, das entspricht 70 % aller Mobilfunkteilnehmer in Kenia [CCK12]. Orange Money ist in 10 Ländern im Einsatz und wird von rund 14 % der Mobilfunkteilnehmer dieser Länder genutzt [Ora12]. Diese Dienste ermöglichen Geschäfte

*Diese Arbeit ist eine gekürzte deutsche Version des Artikels: "Fraud Detection in Mobile Payment Utilizing Process Behavior Analysis", ARES 2013 [RZR+13]

mit elektronischem Geld, welches auch als mMoney bezeichnet wird. Die Benutzer können Bargeld über Distributoren in mMoney konvertieren, um damit Waren bei Händlern zu kaufen, Rechnungen zu bezahlen oder es an andere Nutzer des Dienstes übertragen zu lassen. Wie jeder andere Geldtransferdienst, ist ein MGT-Dienst der Gefahr ausgesetzt, zur Geldwäsche (GW) missbraucht zu werden. Dabei wird eine Verschleierung illegal erwirtschafteter Mittel genutzt, um sie in den legalen Wirtschaftskreislauf einzuschleusen. Es liegt also in der Verantwortung der MGT-Dienstleister, verdächtige GW-Aktivitäten zu erkennen und an die Behörden zu melden. Daraus ergibt sich die Bedeutung von Software-Werkzeugen, wie zum Beispiel von Security Information and Event Management (SIEM)-Systemen, welche Kundendaten zur Laufzeit analysieren und verdächtige Transaktionen erkennen.

In dieser Arbeit wird ein Werkzeug namens Predictive Security Analyzer (PSA) vorgestellt, welches auf einer neuartigen Methode zur prädiktiven Sicherheitsanalyse zur Laufzeit aufbaut [RS10, ER11]. Die Kernidee ist dabei, dass in Ergänzung zu existierenden Verfahren, die statistische Werte aus historischen Messungen verwenden, um Anomalien zu entdecken, hier das Wissen um den geplanten Kontrollfluss der Prozesse verwendet wird. Diese Methode ermöglicht die Auswertung von sicherheitsrelevanten Ereignissen und deren Interpretation in Bezug auf: (1) den spezifizierten Kontrollfluss der Prozesse und (2) die erforderlichen Sicherheitseigenschaften. Bezüglich (1), werden Abweichungen des beobachteten Prozessverhaltens von der vorgegebenen Spezifikation identifiziert. Solche Abweichungen können durch Änderungen in der Prozessausführung, Probleme bei der Messung (z.B., verlorene Ereignisse) oder Anomalien durch Interventionen eines Angreifers verursacht werden. Bezüglich (2) wird eine kontinuierliche Überwachung der für den Prozess festgelegten Sicherheitseigenschaften durchgeführt, um mögliche Sicherheitsverletzungen zu erkennen und vorherzusagen. In diesem Beitrag evaluieren wir die Anwendbarkeit des Ansatzes, um Missbrauchsmuster in den Ereignisströmen der MGT-Transaktionen zu identifizieren, die auf GW-Aktivitäten hinweisen. Dies ist, soweit wir wissen, die erste Anwendung dieser Sicherheitsanalysemethode im MGT-Kontext. Insbesondere zeigen wir, dass der PSA in der Lage ist, (1) einen Ereignisstrom aus einem MGT-System in Echtzeit zu verarbeiten und (2) GW-Warnungen bei betrügerischen Transaktionen zu generieren. Wir beschreiben Ergebnisse von Experimenten mit realen und simulierten Transaktionsprotokollen für verschiedene GW-Szenarien und ermitteln die Sensitivität und Leistungsfähigkeit des PSA.

Das Arbeit ist wie folgt gegliedert: Kapitel 2 stellt das MGT-Szenario und einen Missbrauchsfall im Zusammenhang mit GW vor und Kapitel 3 gibt einen Überblick über den PSA und dessen Anwendung im Experiment. Kapitel 4 beschreibt den Versuchsaufbau und Kapitel 5 präsentiert die experimentellen Ergebnisse. Kapitel 6 gibt einen Überblick über verwandte Arbeiten und Kapitel 7 gibt ein Fazit dieser Arbeit.

2 Missbrauchserkennung in einem System für mobilen Geldtransfer

Diese Arbeit basiert auf einem Anwendungsfall, der in [AGG+11] im Detail beschrieben ist. Dieser Abschnitt beschreibt die wichtigsten Punkte, um den Anwendungsfall zu verstehen. MGT-Systeme sind Systeme, in denen verschiedene Arten von Finanztransaktionen (zum

Beispiel das Ein- und Auszahlen von Bargeld, nationale und internationale Überweisungen, Rechnungszahlungen, etc.) unter Verwendung von mMoney durchgeführt werden. Ein Mobilfunkbetreiber (MFB) emittiert mMoney in Partnerschaft mit einer Privatbank und sendet regelmäßig Complianceberichte an die Zentralbank, die für die Geldpolitik des Landes verantwortlich ist. Das emittierte mMoney kann nur von MFB-Kunden, die den MGT abonniert haben, verwendet werden. Die Teilnehmer sind Endkunden, Dienstleister, Händler und Geschäfte. Diese besitzen ein Prepaid-Konto auf einer MGT-Plattform, das über das Netz des MFB erreichbar ist. Für den Zugriff auf ihre mMoney-Konten verwenden die Endkunden eine dazu passende Anwendung auf ihren mobilen Geräten.

Wir konzentrieren uns hier auf einen Fall von Missbrauch im Zusammenhang mit GW über einen MGT-Dienst wie in Abbildung 1 dargestellt. Die farbigen Pfeile zeigen die regulären Überweisungen, leere Pfeile bezeichnen die betrügerischen. Ein böswilliger Be-

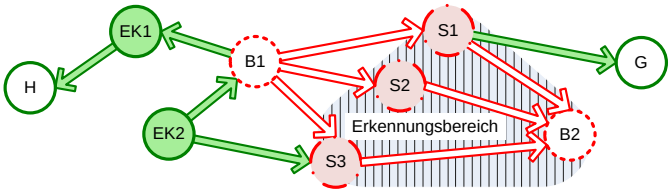


Abbildung 1: MGT Benutzer im Geldwäsche-Szenario: Endkunde (EK), Betrüger (B), Strohmann (S), Händler (H), Geschäft (G)

nutzer (Betrüger) überträgt kleine Geldbeträge auf mehrere Strohmänner. Strohmänner erhalten mMoney von einem Betrüger und überweisen einen hohen Prozentsatz dieses mMoney an einen anderen Betrüger. Jeder Strohmann kann einen kleinen Prozentsatz der Überweisung für sich behalten. Der jeweils letzte Strohmann in einer Kette initiiert den finalen Geldtransfer an einen zweiten betrügerischen Benutzer. Die in dieser Arbeit durchgeführte GW-Analyse basiert auf den in der gestreift markierten Zone dargestellten finalen Geldtransfers. In dieser eingeschränkten Sicht transferiert also ein betrügerischer Benutzer mMoney an einen zweiten betrügerischen Benutzer aber es gibt keine direkten Übertragungsspuren zwischen diesen im MGT-System.

Es gibt eine Vielzahl von GW-Verfahren [FIN12, Int12]. In dieser Arbeit betrachten wir ein GW-Schema mit folgenden Annahmen: (i) es gibt nur einen Strohmann in der Kette der Strohmänner; (ii) der Betrag einer betrügerischen Transaktion ist viel kleiner als der Durchschnitt in diesem Dienst; (iii) Strohmänner führen, außer den betrügerischen Aktionen, auch reguläre mMoney-Überweisungen durch. Diese Annahmen schränken den vorgeschlagenen Ansatz zur Betrugserkennung nicht ein, solange man in der Lage ist, einen Prozessablauf zu einem anderen gewählten GW-Schema zu spezifizieren. Weiterhin hat die Analyse der realen Betriebsprotokolle, die uns vom Betreiber des Dienstes für diese Untersuchung zur Verfügnung gestellt wurden, gezeigt, dass es bei MGT-Benutzern normalerweise keine plötzlichen Änderungen in der Höhe der Transaktionsbeträge gibt.

3 Prädiktive Sicherheitsanalyse zur Laufzeit

Prädiktive Sicherheitsanalyse nutzt eine formalisierte, ausführbare Prozessspezifikation zu einer Berechnung des geplanten Prozessverhaltens im jeweiligen Zustand des überwachten Systems. Während der Laufzeit wird nun das tatsächliche Prozessverhalten aus dem Ereignisstrom des MGT-Systems gemessen und mit dem geplanten Verhalten verglichen, um Anomalien festzustellen. Desweiteren können Sicherheitsanforderungen spezifiziert werden, welche zur Laufzeit am gemessenen Prozessverhalten überprüft werden. Darüber hinaus wird das Wissen über das erwartete Verhalten – aus dem Prozessmodell – verwendet, um Fehler in der nahen Zukunft vorherzusagen. Diese prädiktive Sicherheitsüberwachung ermöglicht es, negativen künftigen Aktionen proaktiv zu begegnen.

Der PSA unterstützt den vollständigen Zyklus dieser ereignisbasierten Sicherheitsanalyse. Die in der MGT-Anwendung verwendeten Funktionen werden im Folgenden beschrieben.

Zunächst extrahiert der PSA Ereignisse aus einem Ereignisstrom oder liest Ereignisse aus einer Datenbank, entsprechend einem vordefinierten Ereignisschema. Basierend auf dem Ereignisschema werden abstrakte Ereignisse definiert, um für die Sicherheitsanalyse nicht relevante Informationen herauszufiltern.

Die vom PSA durchgeführten Simulationen basieren auf formalen Prozessrepräsentationen, welche mittels Asynchroner Produktautomaten (APA) [RRZE14] spezifiziert werden. Ein APA kann als eine Familie von Elementarautomaten angesehen werden, die über gemeinsame Zustandskomponenten kommunizieren. Informale Spezifikationen der Prozesse, die in der Ereignisgesteuerte Prozesskette (EPK)-Notation [KNS92] vorliegen, müssen also entsprechend formalisiert werden, um eine Sicherheitsanalyse durchführen zu können. Die EPK-Notation ist für Anwender, die nicht mit formalen Spezifikationsmethoden vertraut sind, leicht verständlich. Der PSA unterstützt den Benutzer beim Erstellen eines EPK sowie dessen Transformation in ein operationales formales Modell. Existierende EPK-Modelle können unter Verwendung von archivierten Ereignisprotokollen, die als Eingabe benutzt werden, oder auch interaktiv direkt zur Laufzeit angepasst werden. Auf diese Weise wird ein *Uncertainty Management* realisiert, dass eine halbautomatische Anpassung des Prozessmodells, entsprechend den aktuellen Kontextbedingungen, durchführt. Uncertainty-Situationen können insbesondere während der Synchronisierung des Zustands einer laufenden Prozessinstanz mit dem Zustand des Prozessmodells auftreten, wenn das Modell nicht hinreichend exakt oder veraltet ist, unerwartete Ereignisse auftreten oder erwartete Ereignisse nicht empfangen werden. Neben APA-Modellen können Petri Net Markup Language (PNML)-Spezifikationen [WK03], die von Process-Discovery-Werkzeugen generiert werden (z.B., ProM [MMWvdA11]), importiert werden. Für alle Spezifikationsmethoden wird die Berechnung der nächsten Systemzustände auf Modellebene unterstützt.

Sicherheitseigenschaften, die der zu untersuchende Prozess erfüllen soll, werden in Form von Monitorautomaten [RRZE14] spezifiziert. Wird ein Ereignis aus dem überwachten System erkannt, das zu einem Zustandsübergang in einem solchen Monitorautomaten passt, so wird der aktuelle Zustand dieses Automaten entsprechend geändert. Falls ein als kritisch eingestufter Zustand erreicht wird, so wird ein entsprechender Alarm generiert. Falls der PSA in dem Vorhersagehorizont des Prozessverhaltens einen möglichen Zustandsübergang

findet, der zu einem kritischen Zustand in einem Monitorautomaten führt, so wird ein "prädiktiver" Alarm generiert. Der PSA unterstützt die Visualisierung des aktuellen Prozesszustandes und des aktuellen Sicherheitsstatus in Form des Zustands der betroffenen Monitorautomaten.

Der PSA wurde in der vorliegenden Arbeit in zwei Phasen benutzt, einer *Lernphase* und einer *Anomalieerkennungsphase*. In der initialen Lernphase wird das Normalverhalten un-

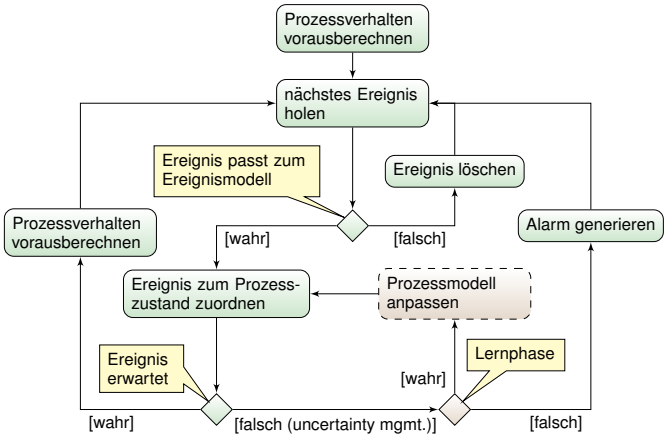


Abbildung 2: Laufzeitverhalten des PSA

ter Verwendung existierender Log-Dateien, die nur korrektes Verhalten enthalten, erlernt. Für diesen Zweck wird eine Abbildung, bei der eine Klassifikation abhängig von der transferierten Geldmenge definiert wird (siehe Tabelle 1), sowie die Reihenfolge der abstrakten Ereignisse aus dem Transaktionsprotokoll verwendet. Beispielweise wird ein Transaktionswert zwischen 500 und 1000 auf den abstrakten Wert $medium$ abgebildet. Diese empirisch erzeugte Abbildung, die auf Transaktionsprotokolle aus dem realen Betrieb des Dienstes angewandt wird, kann für spezifische Trainingsmengen des MGT unterschiedlich sein. Abbildung 2 zeigt eine Übersicht zu den Schritten der Ereignisverarbeitung des PSA in der Lernphase. Die durch eine gestrichelte Linie gekennzeichnete Aktion "Prozessmodell anpassen" wird halbautomatisch durchgeführt und erfordert eine Benutzerinteraktion.

In der Anomalieerkennungsphase werden Abweichungen vom charakteristischen Normalverhalten erkannt und entsprechende Alarme generiert. In unserer experimentellen Testumgebung wurde ein synthetisches Prozessverhalten durch einen Simulator erzeugt, der die Testdaten, basierend auf Eigenschaften von realen Transaktionsdaten, generiert. In der Anomalieerkennungsphase wird die Aktion "Prozessmodell anpassen" in Abbildung 2 nicht verwendet. Stattdessen erfolgt die automatische Generierung von Alarmen durch die Uncertainty-Management-Komponenten des PSA. Um die Anzahl der Fehlalarme zu reduzieren, kann das Modell des Normalverhaltens mit Hilfe von Realdaten, in denen bereits Annotationen für verdächtige Aktionen vorliegen, verbessert werden. Alternativ können entsprechende Filter für die durch den PSA generierten Alarme verwendet werden.

4 Versuchsanordnung

Der PSA wird im nicht-interaktiven Modus verwendet, so dass beim Empfang von unerwarteten Ereignissen automatisch Alarme generiert werden.

Der Transaktionsbetrag ist eine Variable aus $\mathbb{R}$, was eine Diskretisierung erfordert, um eine berechenbare Abstraktion des Verhaltens zu erhalten. Daher wurden empirisch verschiedene Transaktionsbetragsklassen definiert (siehe Tabelle 1).

Tabelle 1: Abbildung zur Diskretisierung des Transaktionsbetrags

Klasse	winzig	sehr klein	klein	normal	mittel	groß	sehr groß	riesig
Betrag	≤ 5	≤ 50	≤ 200	≤ 500	≤ 1000	≤ 2000	≤ 5000	sonst

Prozess Definition: Um die PSA-Analyse durchzuführen, muss der MGT-Prozess mittels EPK definiert werden. Der PSA unterstützt die Möglichkeit mehrere parallel laufende Benutzerprozesse, deren Instanzen das gleiche PSA-Modell zugrunde liegt, mittels dieses Modells zu untersuchen. Hier wurde das allgemeine Verhalten von allen Benutzern verwendet. Da jeder Benutzer völlig frei in der Art der Nutzung des MGT Systems ist (z.B. Auswahl des Betrags, Häufigkeit von Transaktionen, Interessengemeinschaften, etc.), war es eine Herausforderung die entsprechenden Transaktionsabläufe zu definieren. Aus diesem Grund wurde ein Prozess spezifiziert, der aus einem Missbrauchsszenario abgeleitet wurde: Jeder Prozesszustand bezieht sich auf einen bestimmten Transaktionsbetrag, wie in der Diskretisierung definiert. Zustandsübergänge werden durch die korrespondierenden abstrakten Ereignisse angestoßen. Für jeden Zustand sind nur Übergänge in der gleichen Transaktionsbetragsklasse oder zu den benachbarten Transaktionsbetragsklassen autorisiert; alle anderen Übergänge werden als bösartig betrachtet und erzeugen einen Alarm. Der initiale Zustand erlaubt jeden Übergang.

Operationale Logs: Im realen MGT-System existieren verschiedene Arten von Logs (Zugriff, Transfer, etc.). Für unsere Analyse wurden aber nur die Logs von Transaktionen verwendet. Ein Log-Datensatz enthält den Betrag, den Sender und den Empfänger der Transaktion, den Typ des Senders und des Empfängers sowie weitere systemspezifische Felder. Ereignisse, die Log-Einträge generieren, werden durch das Verhalten des Benutzers ausgelöst sobald beliebige Benutzer des Systems Transaktionen durchführen. Im Rahmen des MASSIF-Projektes wurde von Orange für unsere Analyse ein anonymisiertes Log zur Verfügung gestellt, welches 4,5 Millionen Transaktionen enthält, die in einem Zeitraum von 9 Monaten erzeugt wurden. Diese Realdaten wurden benutzt, um eine Echtzeitanalyse des MGT-Datenstroms mittels des PSA durchzuführen. Da keine Gewissheit bzgl. der Ereignisse (d.h., betrügerisch oder normal) besteht, ist eine direkte Betrugserkennung nicht möglich. Um die Erkennungsrate zu untersuchen, wurden daher Simulations-Logs verwendet.

Simulierte Ereignisse: Um Ereignislogs mit sicheren Fehlerraten zu produzieren, wurde ein Missbrauchsfall in einem Simulator [GHA+13] implementiert. Der Simulator modelliert einzelne Trajektorien als eine Folge von Transaktionen mit Beträgen und Zeitintervallen nach Normalverteilung. Die generierten Simulationsdaten basieren auf Eigenschaften, die an realen Beispielen ermittelt wurden und enthalten unterschiedliche Benutzerkategorien aus dem GW-Szenario (siehe Abbildung 1). *Endkunden* führen regelmäßige Transaktionen (Mittelwert 4000, mit einer Standardabweichung von 500), Abhebungen und Einzahlungen durch. *Betrüger* nutzen den Dienst zur Geldwäsche. *Strohmänner* empfangen mMoney ($20 \leq$ Betrag ≤ 100) von einem Betrüger und transferieren den Betrag zu einem weiteren Betrüger unter Einbehaltung von 10% des Betrags. Betrüger und Strohmänner führen ebenfalls regelmäßige Transaktionen durch. *Geschäfte* akzeptieren mMoney als Zahlungsmittel. *Händler* ermöglichen Endkunden, mMoney in Bargeld umzutauschen und umgekehrt. Das GW-Szenario wurde mit den folgenden Parametern konfiguriert:

S1 keine GW: 50 Endkunden, 8 Geschäfte, 4 Händler. Dieser Datensatz wurde verwendet, um zu überprüfen, ob ehrliche Endkunden als Betrüger eingestuft werden.

S2 GW: 50 Endkunden, 5 Strohmänner, 8 Geschäfte, 4 Händler. Dieser Datensatz diente zur Verifikation der Erkennung von Betrugsfällen durch den PSA.

S3 GW mit mehr Beteiligten: 500 Endkunden, 10 Strohmänner, 16 Geschäfte, 4 Händler. Hierbei wurde die Erkennungsrate bei einem kleineren Anteil an betrügerischen Transaktionen bestimmt.

Evaluierungsmetrik: Die vorliegende Analyse betrachtet die Leistungsfähigkeit bzgl. der Laufzeit, sowie der Erkennung von Betrugsversuchen durch den PSA. Es wurde untersucht, ob die Analyse, unter Realzeitbedingungen eines laufenden MGT-Systems, durchgeführt werden kann und ob ein bestimmtes Zeitintervall von der Erkennung bis zur Signalisierung eines Alarms eingehalten werden kann. Die Untersuchungen wurden auf einem Personal Computer (2 Kerne CPU mit 2.70GHz, 4Gb RAM) durchgeführt. Abschließend haben wir die Fehlerrate des PSA gemessen. Die Leistung wird in Anzahl von Ereignissen, die erfolgreich durch den PSA in einer Sekunde bearbeitet werden, gemessen. Für die Erkennungsrate werden unterschiedliche Metriken benutzt: (a) *False Positive*, nicht bösartiges Ereignis wird als bösartig erkannt; (b) *False Negative*, bösartiges Ereignis wird als nicht bösartig erkannt; (c) *True Positive*, bösartiges Ereignis wird als bösartig eingestuft; (d) *True Negative*, nicht bösartiges Ereignis wird als nicht bösartig eingestuft.

5 Experimentelle Ergebnisse

In diesem Abschnitt werden die Ergebnisse der Experimente zusammengefasst.

Rechenleistung: Reale Ereignisse der operationalen Logs des MGT-Systems wurden benutzt, um die Leistungsfähigkeit des PSA zu evaluieren. Prozessinstanzen wurden für

die Kombination aus Benutzerkennung und Transaktionstyp erzeugt. Mit den realen Logs konnte der PSA 640.000 Prozessinstanzen bearbeiten und brauchte 40 Minuten, um die zugehörigen 5,5 Millionen Ereignisse zu verarbeiten und 0,5 Millionen Alarme zu erzeugen. Eine vollständige Analyse auf der gleichen Datenmenge, die keine Alarme produzierte, benötigte 33 Minuten. Im günstigsten theoretischen Fall (keine Alarme werden generiert) kann der PSA mehr als 2.300 Ereignisse / Sek. verarbeiten, während im ungünstigsten Fall die Anzahl sich auf ca. 191 Ereignisse / Sek. reduziert. Im Mittel kann der PSA 100 Millionen Ereignisse an einem Tag auf einem Standard Computer verarbeiten.

Erkennungsrate: Um die Erkennungsrate des PSA zu bestimmen, wurden simulierte Ereignisse verwendet. Aufgrund des stochastischen Charakters der Simulation wurde die Evaluierung mehrmals durchgeführt. Alle Versuche führten zu den gleichen Ergebnissen. Abbildung 3 illustriert die Transaktionen, die von den an Betrugsversuchen Beteiligten des Szenarios S2 durchgeführt wurden (665 Transaktionen). Jeder Knoten repräsentiert einen Benutzer, jede Kante zeigt eine Transaktion, die Kanten werden mit dem Index der Transaktion in der Folge beschriftet. Die grauen Kanten zeigen True Negative, orange (gepunktete) Kanten – False Positive, grüne (dunkelgrau) Kanten bilden True Positive ab, rote (gestrichelte) Kanten – False Negative. Bösartige Transaktionen vom ersten Betrüger zu

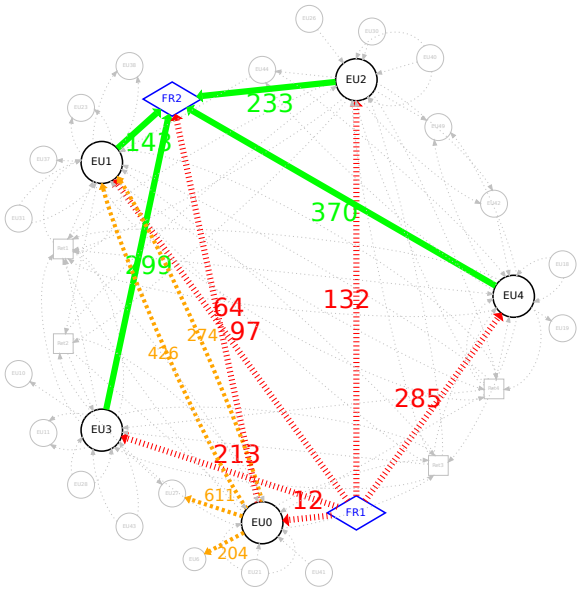


Abbildung 3: Benutzer (EU), Betrüger (FR) und Transaktionen in Szenario S2

den Strohmännern werden nicht als betrügerisch erkannt, was korrekt ist, da die Erkennung solcher Transaktionen nicht intendiert war. Transaktionen von allen Strohmännern, außer $EU0$, zu dem zweiten Betrüger werden korrekt als Betrugsversuch erkannt. Reguläre Transaktionen des Strohmanns $EU0$ werden als bösartig eingestuft und erzeugen falsche Alarme. Tabelle 2 stellt die Sensitivitäts- und Spezifitätskennzahl, die spezifischen Fehler- und Er-

kennungsraten dar. Die Werte in Klammern beziehen sich auf alle GW-Transaktionen ohne Berücksichtigung des gewählten Erkennungsbereichs (siehe Abbildung 1).

Tabelle 2: Erkennungsrate des PSA

	Spezifität	Sensitivität	False Positive Rate	False Negative Rate
S1	100%	–	0%	–
S2	≈ 99,4%	80% (40%)	≈ 0,6%	20% (60%)
S3	≈ 99,9%	90% (45%)	≈ 0,1%	10% (55%)

Da als Basis für die Untersuchung der Erkennungsrate keine realen Ereignisse dienten, kann die tatsächliche Erkennungsrate nicht verifiziert werden. Eine Reduktion der Zahl der Alarme könnte durch eine Verfeinerung der EPK mittels annotierter Realdaten (betrügerisch/nicht betrügerisch) oder mit zusätzlichen Filterkomponenten erfolgen. Das Verhalten des PSA war für alle Szenarien korrekt. Die Fehler bzgl. des Benutzers $EU0$ kommen daher, dass es die erste Transaktion dieses Benutzers ist, mit dem Folgefehler, dass alle nachfolgenden Transaktionen als betrügerisch eingestuft werden. Da das erstellte Prozessmodell nur das veränderte Benutzerverhalten betrachtet, zeigt die zeitliche Abfolge der verdächtigen Transaktionen keine Wirkung auf die Erkennungsfähigkeit des PSA.

Normalerweise wird die Evaluierung von Anomalie-Erkennungswerkzeugen mittels einer ROC-Kurve [Faw04] (spezifiziert durch verschiedene Konfigurierungsschwellwerte $\tau \in \mathbb{R}$) durchgeführt. Der PSA kann nicht mittels solch einfacher Schwellwerte konfiguriert werden. Stattdessen, ist eine komplexe Konfiguration ($\rho = (\rho_{EPC}, \rho_{mapping})$) zusammengesetzt aus einer EPK-Konfigurierung ($\rho_{EPC} \in \mathbb{E}$, $\mathbb{E}$ ist die Menge möglicher EPKs) verbunden mit dem Diskretisierungsschema für Transaktionswerte ($\rho_{mapping} \in \mathbb{M}$, $\mathbb{M}$ ist die Menge aller möglichen Abbildungsfunktionen) erforderlich. Da es schwierig ist verschiedene automatisch generierte Konfigurationen, gegeben durch eine ROC-Kurve, zu durchlaufen, erfolgte die Beschränkung auf einen Punkt.

6 Verwandte Arbeiten

In Bezug auf die Auswertung der Ansätze im Bereich des Geschäftsprozessmanagements in [vdA13], kann man die Funktionalität des PSA-Prototyps als Verfahren zur "Überprüfung der Konformität mit Ereignisdaten" einstufen. In diesem Ansatz werden ein Prozessmodell und Ereignisdaten verwendet, um Abweichungen des Laufzeitverhaltens vom erwarteten Verhalten zu erkennen. Das Interesse für diesen Aspekt des Geschäftsprozessmanagements ist in den letzten drei Jahren gewachsen [vdA13]. Ein ähnlicher Ansatz ist in [RvdA08] beschrieben, aber der Schwerpunkt liegt dort auf einer Quantifizierung von Abweichungen durch die Bildung von Metriken. Wir betrachten die Arbeit bezüglich Laufzeit-Compliance-Überprüfung für Geschäftsprozesse in [MMWvdA11] als Ergänzung zu unserer Arbeit.

Viele Data-Mining-Algorithmen wurden zur Betrugserkennung im Bankenbereich angepasst [DSF12, WLC+13, KMK10]. Filter, Entscheidungsbäume und logistische Regressionsanalyse sind die meist verwendeten Verfahren. Warum eine bestimmte Transaktion als

betrügerisch eingestuft wird, ist mit den Ergebnissen dieser Verfahren leicht zu erklären. Verfahren, die automatisiert ein Modell lernen, werden seltener angewendet. Einige Industrielösungen verwenden jedoch solche Methoden. VISA, beispielsweise implementiert neuronale Netze in deren Betrugserkennungstool RST (Real-Time Scoring) [VIS].

Es gibt mehrere Ansätze Data-Mining-Algorithmen zur Betrugserkennung mittels neuronaler Netze, SVMs, Bayesischen Netzen, Entscheidungsbäumen, angepassten Expertensystemen und Hidden-Markov-Modellen im Umfeld von Kreditkartengeschäften [BJTW11, DAP09]. Für VISA wird in [CGL+07] ein Modell eines mehrdimensionalen Datenwürfels mit separater Änderungserkennung für jede Zelle verwendet. Nach unserer Kenntnis verwenden nicht alle mobilen Zahlungsdienste automatisierte Betrugserkennungslösungen. Die Überwachung kann manuell oder auf Basis von Geschäftsregeln stattfinden. Der M-PESA-Dienst setzt die MinotaurTM Fraud-Management-Lösung basierend auf Geschäftsregeln und neuronalen Netzen ein [Neu]. Nach unserer Kenntnis gibt es keine öffentlichen Arbeiten zur Anpassung der aufgeführten Betrugserkennungsmethoden an MGT-Systeme. Deshalb fällt ein Vergleich unsere Arbeit mit bestehenden Systemen schwer.

7 Fazit

Diese Arbeit nutzt Alarme, welche durch die Uncertainty-Komponente des PSA erzeugt werden, um Aktivitäten, die auf GW hindeuten, in einem MGT-System zu erkennen. Dabei werden GW-Muster in synthetisch generiertem Prozessverhalten, welches auf Basis einer Auswertung der Eigenschaften von Ereignisströmen aus einem realen MGT-System generiert wurde, analysiert. Wir haben gezeigt, dass der PSA in der Lage ist, Betrugserkennung in einem simulierten Szenario durchzuführen. Es konnte gezeigt werden, dass diese Erkennungsleistung effizient ist, aber empfindlich gegenüber Rauschen in einer realen Umgebung reagiert. Es ist daher notwendig, die Rauschfestigkeit durch eine Verbesserung der Korrelation der erzeugten Warnungen oder durch eine spezifische Auswertung des Prozesszustands zu verbessern. Beispielweise kann man in einen kritischen Zustand gehen, wenn die gleiche Warnung mehrfach auftritt. Ergebnisse des PSA sollten von Entscheidungunterstützungs- und Reaktionssystemen ausgewertet werden, um die Sicherheitsregeln des MGT-Systems anzupassen und die betrügerischen Transaktionen automatisch zu blockieren [RCH+12]. Um die Beurteilung des Systems zu erleichtern, wäre es interessant, Methoden zu entwickeln, welche in der Lage sind, eine große Menge von EPKs automatisch zu erzeugen und damit eine breite Basis für weitere Auswertungen bereitstellen.

Danksagung

Die vorliegende Arbeit basiert auf Forschungsergebnissen des Projektes MASSIF (ID 257475), welches durch die Europäische Kommission kofinanziert wurde, sowie des Projektes ACCEPT (ID 01BY1206D), welches durch das Bundesministerium für Bildung und Forschung gefördert wird.

Literatur

[AGG+11] M. Achemlal, S. Gharout, C. Gaber, M. Llanes, E. Prieto, R. Diaz, L. Coppolino, A. Sergio, R. Cristaldi, A. Hutchison und K. Dennie. Scenario requirements. Bericht, MASSIF FP7-257475, 2011.

[BJTW11] S. Bhattacharyya, S. Jha, K. Tharakunnel und J. C. Westland. Data mining for credit card fraud: A comparative study. *Decision Support Systems*, 50, 2011.

[CCK12] CCK. Quarterly sector statistics report. Bericht, Communications Commission of Kenya, 2012.

[CGL+07] C. Curry, R. L. Grossman, D. Locke, S. Vejcik und J. Bugajski. Detecting changes in large data sets of payment card data: a case study. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, Seiten 1018–1022, 2007.

[DAP09] L. Delamaire, H. Abdou und J. Pointon. Credit card fraud and detection techniques: a review. *Banks and Bank systems*, 4, 2009.

[DSF12] R. Dreżewski, J. Sepielak und W. Filipkowski. System supporting money laundering detection. *Digital Investigation*, 9(1):8 – 21, 2012.

[ER11] J. Eichler und R. Rieke. Model-based Situational Security Analysis. In *Workshop on Models@run.time*, Jgg. 794, Seiten 25–36. CEUR, 2011.

[Faw04] T. Fawcett. ROC Graphs: Notes and Practical Considerations for Researchers. *Pattern Recognition Letters*, 27(8):882–891, 2004.

[FIN12] FINTRAC Typologies and Trends Reports. Money Laundering and Terrorist Financing Trends in FINTRAC Cases Disclosed Between 2007 and 2011. `http://www.fintrac-canafe.gc.ca/publications/typologies/2012-04-eng.asp#s1-1`, April 2012. Last visit on 21/05/2013.

[GHA+13] C. Gaber, B. Hemery, M. Achemlal, M. Pasquet und P. Urien. Synthetic logs generator for fraud detection in mobile transfer services. In *Proceedings of the 2013 International Conference on Collaboration Technologies and Systems (CTS2013)*, 2013.

[Int12] Internal Revenue Service (IRS). Examples of Money Laundering Investigations. Fiscal Year 2012. `http://www.irs.gov/uac/Examples-of-Money-Laundering-Investigations-Fiscal-Year-2012`, October 2012. Last visit on 21/05/2013.

[KMK10] N. A. L. Khac, S. Markos und M.-T. Kechadi. A Data Mining-Based Solution for Detecting Suspicious Money Laundering Cases in an Investment Bank. In *Advances in Databases Knowledge and Data Applications (DBKDA), 2010 Second International Conference on*, Seiten 235–240, 2010.

[KNS92] G. Keller, M. Nüttgens und A.-W. Scheer. Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)". *Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes*, 89, 1992.

[MMWvdA11] F. M. Maggi, M. Montali, M. Westergaard und W. M. P. van der Aalst. Monitoring Business Constraints with Linear Temporal Logic: An Approach Based on Colored Automata. In *Business Process Management (BPM 2011)*, Jgg. 6896 of *LNCS*, Seiten 132–147. Springer, 2011.

[Neu] Neural technologies. MinotaurTM Fraud Detection Software - Finance Sector. http://www.neuralt.com/fraud_detection_software.html. Last visited on 23/03/2013.

[Ora12] Orange. Orange Money. http://www.orange.com/en/press/press-releases/press-releases-2012/Orange-Money-reaches-4-million-customers-and-launches-in-Jordan-and-Mauritius, June 2012. Last visit on 12/04/2013.

[RCH+12] R. Rieke, L. Coppolino, A. Hutchison, E. Prieto und C. Gaber. Security and Reliability Requirements for Advanced Security Event Management. In I. Kotenko und V. Skormin, Hrsg., *Computer Network Security*, Jgg. 7531 of *Lecture Notes in Computer Science*, Seiten 171–180. Springer Berlin Heidelberg, 2012.

[RRZE14] R. Rieke, J. Repp, M. Zhdanova und J. Eichler. Monitoring Security Compliance of Critical Processes. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22th Euromicro International Conference on*. IEEE Computer Society, 2014.

[RS10] R. Rieke und Z. Stoynova. Predictive Security Analysis for Event-Driven Processes. In *Computer Network Security*, Jgg. 6258 of *LNCS*, Seiten 321–328. Springer, 2010.

[RvdA08] A. Rozinat und W. M. van der Aalst. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64 – 95, 2008.

[RZR+13] R. Rieke, M. Zhdanova, J. Repp, R. Giot und C. Gaber. Fraud Detection in Mobile Payment Utilizing Process Behavior Analysis. In *Proceedings of 2013 International Conference on Availability, Reliability and Security, ARES 2013*, Seiten 662–669. IEEE Computer Society, 2013.

[vdA13] W. M. P. van der Aalst. Business Process Management: A Comprehensive Survey. *ISRN Software Engineering*, Seite 37, 2013.

[VIS] VISA. Security and trust at every level. http://www.visaeurope.com/en/about_us/security.aspx. Last visit on 22/03/2013.

[WK03] M. Weber und E. Kindler. The Petri Net Markup Language. In *Petri Net Technology for Communication-Based Systems*, Jgg. 2472 of *LNCS*, Seiten 124–144. Springer, 2003.

[WLC+13] W. Wei, J. Li, L. Cao, Y. Ou und J. Chen. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, 16(4):449–475, 2013.

Zertifizierte Datensicherheit für Android-Anwendungen auf Basis statischer Programmanalysen

Steffen Bartsch[1], Bernhard J. Berger[2], Eric Bodden[3], Achim D. Brucker[4], Jens Heider[3], Mehmet Kus[5], Sönke Maseberg[6], Karsten Sohr[2], Melanie Volkamer[1]

[1]Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt
[2]Universität Bremen, Bibliothekstr.1, 238359 Bremen
[3]Fraunhofer SIT, Rheinstraße 75, 64295 Darmstadt
[4]SAP AG, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe
[5]OTARIS Interactive Services GmbH, Fahrenheitstraße 7, 28359 Bremen
[6]datenschutz cert GmbH, Konsul-Smidt-Straße 88, 28217 Bremen

Abstract: Smartphones erfreuen sich einer stetig wachsenden Beliebtheit. Ein Grund hierfür ist die Vielzahl verschiedenster mobiler Anwendungen. Mit den Chancen, die sich hierdurch für den Benutzer, aber auch Organisationen bieten, sind Risiken verbunden, die beispielsweise zu einer Verletzung der Privatsphäre führen können. In diesem Beitrag diskutieren wir, wie statische Programmanalyse dabei helfen kann, Android-Anwendungen bzgl. der Sicherheit zu zertifizieren.

1 Einleitung

Smartphones finden eine immer weitere Verbreitung sowohl im privaten als auch im geschäftlichen Umfeld; laut Bitkom besitzt jeder dritte Deutsche ein Smartphone – Tendenz steigend. Wesentlich für den Erfolg von Smartphones ist die Möglichkeit, mobile Anwendungen (sog. „Apps“) bequem aus Markets zu installieren. Das Angebot im Android Market „Google Play“ umfasst inzwischen mehr als 1 Mio. mobile Anwendungen, die insgesamt über 25 Mrd. Mal heruntergeladen wurden. In jedem Monat kommen ca. 30.000 neue Anwendungen dazu. Auch in speziellen Markets für Geschäftskunden, wie er z.B. von der SAP AG betrieben wird, nimmt die Anzahl von Geschäfts-Apps rasant zu. Mit den Chancen dieser Entwicklung gehen jedoch große Risiken einher, vor allem durch die wachsende Zahl mobiler Anwendungen mit oft unbekannter Herkunft. Hierdurch steigt die Gefahr der Verbreitung von Schadsoftware, die sich beispielsweise als nützliche Anwendung tarnt [Ge11]. Zudem können Schwachstellen in Apps von Angreifern als Einstiegspunkte genutzt werden, um Zugriff auf Firmendaten zu erhalten. Dass Schwachstellen verstärkt in Apps auftreten, zeigen die aktuellen Berichte über Sicherheitsprobleme des WhatsApp-Messengers und die fehlerhafte Implementierung der SSL-Verschlüsselung quer durch einen großen Anteil von sicherheitskritischen Apps [Fa12].

Aufgrund dieser Entwicklung ist es erforderlich, das Thema „Sicherheitsanalyse von mobilen Anwendungen“ **grundlegend und umfassend** zu bearbeiten und eine Analyse- und Zertifizierungsplattform für Apps zu entwickeln. Nachfolgend skizzieren wir, wie eine solche Plattform aussehen könnte, und beschreiben einen statischen Analyse-Ansatz

für Android Apps, der als Basis der Zertifizierungsplattform dienen kann. Wir diskutieren unseren Ansatz anhand von Android, da dieses aktuell die am weitesten verbreitete Smartphone-Plattform ist.

Die eigentlich wissenschaftlich spannende und herausfordernde Aufgabe für eine Sicherheitsanalyse von Android-Apps ergibt sich daraus, dass Android als ein komplexes verteiltes System aufgefasst werden kann, das aus vielen verschiedenen Apps und Komponenten besteht (Multi-Applikationen-System). Die Apps können auf diversen Wegen miteinander kommunizieren oder auf einander zugreifen, insbesondere über Nachrichten (Interprozess-Kommunikation) oder gemeinsam genutzte Speicherbereiche wie z.B. externe Speicherkarten [En11]. Hierbei können für den Benutzer unerwünschte Informationsflüsse entstehen, wenn die Apps gewollt oder unabsichtlich fehlerhaft programmiert worden sind. Durch die Notwendigkeit, mehrere Betriebssysteme (z.B. Android, iOS, Windows Phone) zu unterstützen, nimmt die Zahl der Apps, welche plattformunabhängige JavaScript-Anteile mit plattformabhängigen Java (Android) oder Objective-C (iOS) Anteilen kombinieren, zu. Gerade im Zusammenspiel dieser unterschiedlichen Technologien entstehen bisher unbekannte Möglichkeiten für Verletzungen der Datensicherheit. Auch die unterschiedlichen Kommunikationswege eines Smartphones mit der Außenwelt wie z.B. Internet, SMS/MMS, Bluetooth oder Near Field Communication (NFC) machen die Aufgabenstellung komplexer, aber auch interessanter. Da Android eine Multi-Applikationen-Plattform ist, werden Analysen benötigt, die die Kommunikationsstruktur mehrerer Apps analysieren und mit Informationsflussergebnissen für einzelne App-Komponenten kombinieren können, um Lecks privater Daten zuverlässig zu erkennen.

Bei Android-Apps liegt es grundsätzlich im Ermessen des Benutzers, ob eine von der App geforderte Berechtigung angemessen für den Zweck der App ist. Gleiches nehmen wir für die Ergebnisse der innovativen Sicherheitsanalysen der Apps an: Die beste Analyse ist in der Praxis hinfällig, wenn ihre Ergebnisse nicht verstanden werden. Ziel ist hierbei, es auch Sicherheitsadministratoren in Unternehmen oder technisch versierten Benutzern zu ermöglichen, die Analyseergebnisse direkt zu interpretieren. Auf Basis der Analysen können sie dann fundiert die Entscheidung treffen, ob eine App installiert werden darf.

Eine weitere Herausforderung besteht in einem leichtgewichtigen Prozess zur Sicherheitszertifizierung von Software, der der Dynamik der Entwicklung von Android-Apps Rechnung trägt; ein solcher Prozess existiert weltweit noch nicht. Evaluationsschemata wie die Common Criteria sind hierfür zu aufwendig. Perspektivisch bietet sich auch eine Anwendung im Bereich der Zertifizierung von Java-basierter Software von kleinen und mittelständischen Unternehmen (KMU) an, die sich oft eine aufwendige Evaluation wie bei den Common Criteria nicht leisten können.

Der Rest dieses Beitrages ist folgendermaßen gegliedert. Abschnitt 2 diskutiert eine Zertifizierungsplattform für Android-Anwendungen, während die Notwendigkeit von Sicherheitsanalysen als Basis für die Zertifizierungsplattform Gegenstand von Abschnitt 3 ist. In Abschnitt 4 stellen wir unseren Lösungsansatz für einen leichtgewichtigen Zertifizierungsprozess für Apps vor. In Abschnitt 5 geben wir ein Fazit und einen Ausblick.

2 Zertifizierungsplattform für mobile Anwendungen

Abbildung 1 gibt eine Übersicht einer denkbaren Zertifizierungsplattform. Insbesondere wird hier folgendes Szenario unterstützt, das die Lieferkette zur Bereitstellung von Apps abdeckt: Der App-Hersteller lässt sich die Sicherheit seiner App von einer Zertifizierungsstelle auf Grundlage der Prüfung eines Prüflabors bestätigen, das ein entsprechendes Sicherheitswissen über mobile Anwendungen besitzt. Das Prüflabor führt Prüfungen gemäß einem an die App-Dynamik angepassten Zertifizierungsverfahren durch, das sowohl auf statischen als auch dynamischen Analysen beruht. Bei Unbedenklichkeit der App (gemäß vorgegebenen Zertifizierungskriterien/Policy) wird ein Zertifikat vergeben und in die Zertifizierungsplattform eingespielt, das automatisiert ausgewertet werden kann. Marktbetreiber (insbesondere spezieller App-Märkte) wie z.B. die SAP AG erhalten durch die Plattform Zugriff auf das Zertifikat und können automatisiert beim Hochladen einer App in den Market prüfen, ob diese den Sicherheitsrichtlinien des Marktes entspricht und ggf. Nachbesserungen verlangen. Ferner können Werkzeuge von Nutzen sein, die den App-Hersteller aktiv bei der Korrektur gefundener Sicherheitslücken unterstützen.

Im Gegensatz zur beschriebenen automatischen Auswertung der Zertifikate besteht zusätzlich die Möglichkeit, manuelle Überprüfungen vorzunehmen. Dies erfordert eine verständliche Darstellung der Analyseergebnisse. Ziel ist hierbei, auch technisch versierten Benutzern oder Sicherheitsadministratoren in Unternehmen zu ermöglichen, die Analyseergebnisse direkt zu interpretieren. Auf Basis der Analysen können sie dann fundiert die Entscheidung treffen, ob eine App installiert werden darf.

Durch die Anbindung der Zertifizierungsplattform an Device Management-Lösungen werden ferner Großunternehmen unterstützt, die Sicherheitsrichtlinien auf einer Vielzahl von mobilen Endgeräten umsetzen müssen. Basis einer solchen Zertifizierung sind die Sicherheitsanalysen der Apps. Hier bieten sich statische Analysen an, die in den letzten Jahren immer ausgereifter geworden sind [CW07]. Statische Analysen werden vor der Ausführung eines Programmes wie z.B. einer App durchgeführt und nehmen den Quellcode oder auch den Binärcode als Gegenstand der Untersuchungen. Prinzipiell sind das ähnliche Techniken und Datenstrukturen, wie sie beim Kompilieren und Übersetzen von Programmen eingesetzt werden. Dynamische Analysen werden im Gegensatz dazu bei der Ausführung des Programmes durchgeführt. Nachfolgend geben wir einen Überblick über bereits vorhandene statische und dynamische Sicherheitsanalysen für Android-Apps und leiten den Bedarf für zukünftige Analysen ab.

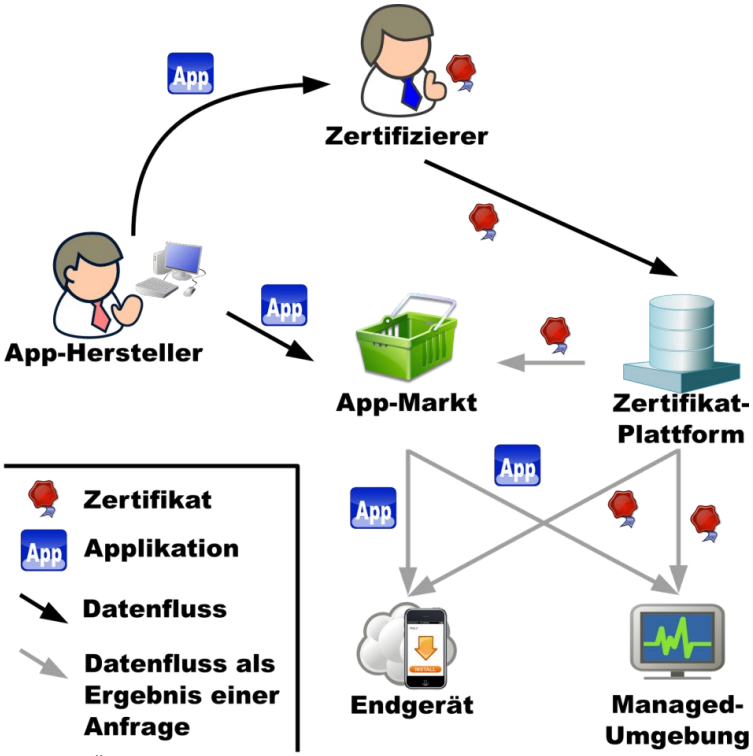


Abbildung 1: Übersicht über die Zertifizierungsplattform und die verschiedenen Rollen im Zertifizierungsprozess.

3 Sicherheitsanalysen von Android-Anwendungen

Google nimmt bereits erste Überprüfungen von Apps in Google Play mittels des Werkzeugs „Bouncer“ vor [Go12]. Es ist aber nicht transparent, welche Überprüfungen in welcher Tiefe durchgeführt werden. Kürzlich zeigte sich, dass Malware-Apps sich der Erkennung durch den Bouncer entziehen können, indem sie erkennen, wenn der Bouncer sie analysiert und sich zu diesem Zeitpunkt harmlos geben [Fi12] – ein Nachteil rein dynamischer Analysen. In Android 4.2 wurde zudem ein App Verification-Dienst eingeführt. Wie sich jedoch schnell zeigte, ist dieser äußerst unbrauchbar und kann keinerlei Sicherheitsgarantien geben [Ji12]. Der Dienst berechnet lediglich zu einer zu installierenden App einen Hash und gleicht diesen dann mit den Hashes bekannter Malware-Apps ab. Der Hash ist jedoch zu statisch: Selbst leichte (automatisierbare) Änderungen am Code der Malware führen dazu, dass diese nicht mehr erkannt wird.

Sicherheitsanalysen mit der Entwicklung eines machbaren Konzeptes zur Zertifizierung von mobilen Anwendungen existieren bislang nicht am Markt. Vor allem die Interprozess-Kommunikation (IPC), mit der Android-Anwendungen Daten über Prozessgrenzen hinweg austauschen können, lässt sich nicht mittels statischen Analysewerkzeugen, wie derzeit am Markt vorhanden, adressieren. Android-Applikationen setzen sich gemäß Android-Programmiermodell aus verschiedenen Komponenten zusammen. Android-

Komponenten sind u.a. Activities (verantwortlich für die graphische Darstellung und User-Interaktion) und Services (welche die eigentliche Hintergrundarbeit leisten); Broadcast Receiver können Broadcast-Nachrichten abonnieren und darauf reagieren. Diese Komponenten kommunizieren via IPC über sogenannte Intents (spezielle Datenstrukturen zum Nachrichtenaustausch). Eine Analyse muss verstehen, wo ein Intent gesendet wird und welche Callbacks den Intent empfangen. Dies benötigt eine sehr präzise statische Programmanalyse, um komponentenübergreifende Informationsflüsse berechnen zu können. Ferner muss berücksichtigt werden, dass Android eine ganze Reihe solcher IPC-Mechanismen (beispielsweise ein Callback für den Klick eines Buttons) bereitstellt, die allesamt von Analysen unterstützt werden müssen. Mithin sind Standardwerkzeuge wie IBM AppScan [Ib13] und Fortify SCA [Fo13] für solche Analysen nicht geeignet.

Im Forschungsumfeld gibt es Analyse-Werkzeuge, die auf Android zugeschnitten sind. Chin et al. berücksichtigen in ihrem ComDroid-Werkzeug zwar die Interprozess-Kommunikation, führen aber nur einfache statische Analysen direkt auf dem Android-Binär-Code (DEX-Code) aus [Ch11] und können somit keine ausgereiften Zeigeranalysen verwenden, wie sie in Java-/Bytecode-Analyseframeworks wie z.B. Soot [Va99] implementiert worden sind. Daher ist die Analyse nicht sehr präzise. Grace et al. analysieren System-Apps, die z.B. vom Geräte-Hersteller mitgeliefert werden, mit Hilfe ihres Woodpecker-Werkzeugs auf Sicherheitslücken [Gr12]; da sie nur den Control Flow Graph (CFG) für ihre Analyse verwenden, unterstützen sie keine ausgereiften Datenflussanalysen. Lu et al. haben ein statisches Analyse-Verfahren entwickelt, das Android-Apps auf sog. „Confused Deputy“-Probleme hin untersucht und die Framework-Konzepte von Android berücksichtigt [Lu12]. Confused Deputy-Lücken erlauben es einer bösartigen Anwendung auf eine andere App „aufzuspringen“ und kritische Berechtigungen (z.B. Zugriff auf Telefonlisten, Kontaktdaten) auszuführen, obwohl der Angreifer diese Berechtigungen nicht besitzt.

Die eben genannten Werkzeuge sind meist nur als Proof-of-Principles, teilweise durch verschiedene Skripte implementiert, vorhanden. Oft behandeln sie auch nur einen bestimmten Teilaspekt der Android-Sicherheit. Die verschiedenen Werkzeuge untermauern jedoch noch einmal die Relevanz und Aktualität des Themas „Statische Analyse von Android-Apps“, auch wenn es sich eher um Ad hoc-Lösungen handelt, die insbesondere nicht den Zertifizierungsaspekt von Apps und die damit verbundenen Prozesse berücksichtigen. Zudem ist keines der bisher verfügbaren Werkzeuge in der Lage, Informationsflüsse zwischen JavaScript- und Java-Komponenten zu analysieren.

4 Lösungsansatz

Die Neuheit unseres Lösungsansatzes besteht in der Analyse der Interprozess-Kommunikation auf Basis von speziellen, auf Android-Apps zugeschnittenen statischen Code-Analysen mit der Entwicklung eines an die App-Dynamik angepassten Zertifizierungskonzeptes. Statische Analyseverfahren für diese Kommunikation sind derzeit zwingend erforderlich, jedoch nicht existent.

Eine weitere wesentliche Neuerung unseres Analyseansatzes besteht darin, für einzelne Android-Anwendungen (und Android-Komponenten) zuerst Analysen separat durchzuführen und die Einzelergebnisse zu „Summaries“ zusammenzufassen. Diese Summaries werden anschließend unter Berücksichtigung der Interprozesskommunikation sowie der Kommunikation zwischen Komponenten unterschiedlicher Technologien (z.B. JavaScript und Java) zu einem Gesamtergebnis zusammengesetzt (Komposition der Einzelergebnisse). Des Weiteren werden die Analysen so gestaltet, dass sie einem Anwender genügend Informationen liefern, um die Analyse-Ergebnisse nachvollziehen und interpretieren zu können (Benutzbarkeitsaspekt). Als Basiswerkzeug für die Analysen kann beispielsweise das bereits oben genannte Soot-Werkzeug verwendet werden. Wir haben bereits einen Prototyp entwickelt, der auf Soot aufsetzt und Analysen unterstützt, die die Android-Framework-Mittel berücksichtigen und auf Basis des Android-Binärcodes (DEX) arbeiten [Fr13]. Auf dieser Basis werden wir weitere Komponenten für eine umfassende App-Analyse entwickeln wie z.B. eine verständliche Darstellung der Analyse-Ergebnisse in Form von architekturellen Beschreibungen.

Im Folgenden beschreiben wir kurz die einzelnen Schritte unseres Analyse-Ansatzes:

1. Wandele den DEX-Code einer oder mehrerer Android-Anwendungen in das Zwischenformat des Soot-Werkzeuges um.
2. Erzeuge mit Hilfe von Soot eine abstrakte Form des Programms, welche die Software-Architektur wie z.B. die einzelnen Android-Komponenten, deren Einstiegs- und Ausstiegspunkte, IPC-Verbindungen sowie Call-Graph-Informationen enthält.
3. Ermittle auf dem Soot-Zwischenformat mit Hilfe von Datenflussanalysen Informationsflüsse zwischen Einstiegs- und Austrittspunkten.
4. Fasse die ermittelten Informationsflüsse pro Komponente zusammen (Summaries).
5. Berechne mit Hilfe dieser Summaries auf der in Schritt 2 erzeugten abstrakten Darstellung alle Informationsflüsse zwischen mehreren Komponenten (bzw. mehreren Anwendungen) und ermittele Informationsflüsse, die vertrauliche Daten nach außen leiten (z.B. Versenden von Kontaktdaten in das Internet oder per MMS).
6. Stelle die Analyseergebnisse in einer für Sicherheitsanalysten verständlichen Form dar und überprüfe, ob die Informationsflüsse durch einzelne Anwendungen bzw. die Kombination von Anwendungen der definierten Policy entsprechen.

Zudem ist es vorgesehen, zusätzliche Analysen für in JavaScript entwickelte Komponenten zu integrieren, so dass in Schritt 5 auch Datenflüsse in hybriden Apps untersucht werden können.

Die Analyseschritte können z.T. getrennt voneinander ausgeführt werden. Dadurch ist es möglich, dass die aufwendigen Schritte (vor allem die Zeigeranalysen in Schritt 3) einmalig durchgeführt werden und ein Nutzer der Analyse nur noch weniger aufwendige Schritte (vor allem 5 und 6) vornehmen muss. Insbesondere werden die Schritte 5 und 6 direkt auf dem Mobiltelefon nutzbar sein. Dies ist wichtig, weil das Schadpotenzial einer App oft vom Vorhandensein anderer Apps oder einer bestimmten Version des Frameworks abhängt. Diese Informationen liegen im App Market nur unzureichend vor. Eine schnelle abschließende Analyse auf dem Smartphone kann hier für Gewissheit sorgen. Die automatische statische Analyse wird somit in Zweifelsfällen durch dynamische Analysen ergänzt, um die Ergebnisse präziser zu machen. Unternehmen können sich

damit ohne eine aufwendige Analyse auf die Angabe der vorhandenen Informationsflüsse verlassen.

Schritt 6 bewirkt eine verständliche Darstellung der durch die statischen und dynamischen Analysen ermittelten Ergebnisse. Eine Möglichkeit besteht darin, diese Ergebnisse in Form von Datenflussdiagrammen (DFDs) darzustellen. DFDs werden von Microsoft beim Threat Modeling, einer Sicherheitsanalyse auf Basis der Software-Architektur, verwendet und sind recht weit verbreitet [He06]. Hierdurch können auf Ebene der Software-Architektur kritische Informationsflüsse von Datenquellen (Lokalisierungsdaten, Kontaktlisten, Telefonlisten) zu Datensenken (wie z.B. dem Internet) in übersichtlicher Form dargestellt werden. Ggf. sind die DFDs noch um spezielle Elemente zu erweitern, die Android-spezifische Framework-Mittel berücksichtigen.

4.1 Zertifizierungsprozess für Android-Anwendungen

Die Ergebnisse zu existierenden Informationsflüssen können für eine Zertifizierung verwendet werden, um sich von der Unbedenklichkeit einer Applikation zu vergewissern. Obwohl sich die Evaluierung und Zertifizierung mittels Common Criteria [Co09] oder gemäß dem Datenschutz-Gütesiegel für eine ganze Reihe von IT-Produkten bewährt hat, ist ein Zertifizierungsprozess für Smartphone-Apps derzeit nicht vorhanden. Zudem existiert kein Zertifizierungskonzept, bei dem Smartphones kontrollieren können, nur zertifizierte Apps zu installieren. Von daher ist dieser Ansatz ein neuartiger und aus Benutzersicht und aus Sicht einer Institution, die Smartphones an ihre Mitarbeiter ausgibt, notwendiger Ansatz. Da der Erfolg des Apps-Konzeptes gerade darin liegt, dass es sich hier oft um Low-Cost-Produkte handelt, ist ein abgestuftes Zertifizierungskonzept wünschenswert, mit einer geringen Prüftiefe (evtl. in Form eines automatischen Dienstes) bis hin zu einer detaillierteren manuellen Analyse für besonders sicherheitskritische und weit verbreitete Apps.

Ein Prüf- und Zertifizierungsschema für mobile Anwendungen auf Basis von statischen und dynamischen Analysen sollte insgesamt folgende Aspekte umfassen:

- den inhaltlichen Umfang der Prüfung auf Basis der Ergebnisse der Projektpartner mit dem zu entwickelnden Analysewerkzeug,
- die formalen Aspekte zur Prüfung und Zertifizierung,
- die Einbeziehung der Methoden internationaler und nationaler Kriterienwerke im Bereich von Datensicherheit,
- ein Validierungssystem zur automatischen Auswertung der Gültigkeit von Zertifikaten,
- den Aufbau von Prüf- und Zertifizierungsstelle mit allen relevanten Prozessen, im Einklang mit anerkannten Standards.

4.2 Verständliche Ergebnisse

Um in der Praxis Anwender tatsächlich zu unterstützen, werden die komplexen Analyseergebnisse und App-Bewertungen verständlich aufbereitet. Sowohl Benutzer ohne oder mit beschränktem technischen Verständnis als auch Sicherheitsexperten sollen die

Ergebnisse einschätzen und basierend darauf fundierte Entscheidungen zu ihrer Sicherheit fällen können. Die Aufbereitung erfolgt in Form von jeweils dem Benutzer angepassten Darstellungen, die das Vorwissen, die mentalen Modelle und die Ziele der Anwender einbeziehen. Es wird unter anderem in Benutzerstudien geprüft, ob die Sicherheit anhand der Darstellung verständlich wird.

5 Fazit

Zusammenfassend betrachtet besteht die Neuheit des vorgeschlagenen Lösungsansatzes darin, ausgereifte Methoden der Programmanalyse im Kontext der Multiapplikationen-Plattform Android anzuwenden. Hierbei wird insbesondere die Semantik des Android-Frameworks mit einbezogen. Einen leichtgewichtigen Zertifizierungsprozess für mobile Anwendungen auf Basis von fundierten Sicherheitsanalysen zu schaffen ist eine herausfordernde, aber dennoch wichtige Aufgabe, um das Vertrauen in IT-Systeme zu stärken. Wir werden diese Aufgabe im Rahmen eines Forschungsprojektes mit einem Konsortium grundlegendend und umfassend bearbeiten, zumal wir die komplette Lieferkette für die Bereitstellung von sicheren und datenschutzgerechten Apps abdecken, von App-Entwicklern, über Prüfstellen bis hin zu Marktplatzbetreibern. Die Forschungspartner werden in diesem Kontext die benötigten Analyseverfahren konzipieren und umsetzen. Wir sind auch optimistisch, dass wir einen grundsätzlichen Beitrag zu einer leichtgewichtigen Zertifizierung von Software liefern können. Gerade kleinere Software-Hersteller können hiervon profitieren.

Danksagung

Diese Arbeit wurde vom Bundesministerium für Bildung und Forschung gefördert (ZertApps-Projekt). Weitere Informationen können Sie unter www.zertapps.de finden.

Darüber hinaus möchten wir uns bei den anonymen Gutachtern bedanken, die wesentlich dazu beigetragen haben, diesen Beitrag zu verbessern.

Literatur

[Co09] Common Criteria: Common Criteria for Information Technology Security Evaluation—Part 1: Introduction and general model, 2009. Zugreifbar unter: http://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R3.pdf

[Ch11] E. Chin, A. Porter Felt, K. Greenwood, D. Wagner. Analyzing Inter-Application Communication in Android. In Proceedings of the 9th International Conference on Mobile Systems, Applica-tions, and Services (MobiSys '11). ACM, USA, 239-252, 2011

[CW07] B. Chess., J, West. Secure Programming with Static Analysis. Addison-Wesley, 2007.

[En11] W. Enck, D. Octeau, P. McDaniel, S. Chaudhuri. A Study of Android Application Security. In Proceedings of the 14th USENIX Security Symposium, August 2011.

[Fa12] S. Fahl, M. Harbach, T. Muders, M. Smith, L. Baumgärtner, B. Freisleben. Why Eve and Mallory love Android: An analysis of Android SSL (in)security. In: Proceedings of the 2012 ACM conference on Computer and communications security.pp. 50–61. CCS ’12, USA , 2012.

[Fi12] D. Fisher. Researchers Find Methods for Bypassing Google's Bouncer Android Security. 2012. Zugreifbar unter: https://threatpost.com/researchers-find-methods-bypassing-googles-bouncer-android-security-060412/76643

[Fo13] Fortify Software. Website, 2013. Zugreifbar unter: http://www.fortify.com.

[Fr13] C. Fritz, S. Arzt, S. Rasthofer, E. Bodden, A. Bartel, J. Klein, Y. le Traon, D. Octeau, P. McDaniel. Highly Precise Taint Analysis for Android Applications, Technical Report TUD-CS-2013-0113, EC SPRIDE, 2013.

[Ge11] Geek.com: Google Yanks Android Apps From Market Over Malware Concerns. http://www.geek.com/articles/mobile/google-yanks-android-apps-from-market-over-malware-concerns-2011032/

[Go12] Google Inc.: Google Mobile Blog – Android and Security. Februar 2012. Zugreifbar unter: http://googlemobile.blogspot.de/2012/02/android-and-security.html

[Gr12] M. Grace, Y. Zhou, Z. Wang, X. Jiang. Systematic Detection of Capability Leaks in Stock Android Smartphones. Proceedings of the 19th Network and Distributed System Security Symposium (NDSS 2012). San Diego, CA, February 2012.

[He06] S. Hernan, S. Lambert, T. Ostwald, A. Shostack. Uncover security design flaws using the STRIDE approach. MSDN Magazine, Nov 2006. Zugreifbar unter http://msdn.microsoft.com/en-us/magazine/cc163519.aspx

[Ib13] IBM Inc. AppScan-Website. 2013. Zugreifbar unter: http://www01.ibm.com/software/rational/products/appscan/source/.

[Ji12] W. Jiang. An Evaluation of the Application (“App”) Verification Service in Android 4.2. North Carolina State University, 2012.

[Lu12] L. Lu, Z. Li, Z. Wu, W. Lee, W., G. Jiang. CHEX: statically vetting Android apps for component hijacking vulnerabilities. In: Proc. of the 2012 ACM Conference on Computer and Communications Security. pp. 229–240. CCS ’12, 2012.

[Va99] R. Vall´ee-Rai, P. Co, E. Gagnon, L. Hendren, P. Lam, V. Sundaresan. Soot - A Java Bytecode Optimization Framework. In CASCON ’99: Proceedings of the 1999 Conference of the Centre for Advanced Studies on Collaborative Research. IBM Press, 1999.

Quantitative Ansätze zur IT-Risikoanalyse

Erik Tews, Christian Schlehuber

Security Engineering Group
Technische Universität Darmstadt
Hochschulstr. 10
D-64289 Darmstadt
e_tews@seceng.informatik.tu-darmstadt.de
cschlehuber@seceng.informatik.tu-darmstadt.de

Abstract: Im Bereich der Risikoanalyseverfahren für kritische IT-Systeme werden primär qualitative Analyseverfahren eingesetzt, welche auf einer Expertenschätzung des Risikos beruhen. Dieses Vorgehen birgt ein hohes Risiko für Fehleinschätzungen durch den Analysten, welches im schlimmsten Fall zu einem unbrauchbaren Sicherheitskonzept führen kann, da die Sicherheitsarchitektur entsprechend der anfänglichen Risikoanalyse gestaltet wird. Diese Arbeit stellt ein erweitertes Verfahren vor, welches durch feingranulare Schätzungen von einzelnen Faktoren die Wahrscheinlichkeit eines Fehlers minimiert und auf diese Weise einen weiteren Schritt in Richtung quantitative Methoden zur IT-Risikoanalysen beschreitet. Zu diesem Zweck kommen Bewertungsskalen zum Einsatz, welche eine spätere Berechnung von Scores ermöglichen. Hierfür wird auf den Assets der jeweiligen Domäne beginnend eine systematische Analyse von möglichen Zielen, Angreifergruppen, Angreifermitteln und denkbaren Schwachstellen durchgeführt. Die Angreifergruppen sowie die möglichen Mittel werden feingranular modelliert und in einer abschließenden Bewertung auf ihr potentielles Risiko für das System überprüft.

1 Einleitung

IT-Systeme werden in vielen sicherheitskritischen Bereichen sowie in kritischen Infrastrukturen eingesetzt. Im Rahmen dieser Verwendung muss folglich die Sicherheit von solchen Systemen sichergestellt werden, was auch durch die Politik im Rahmen des aktuellen Referentenentwurfs „Zur Erhöhung der Sicherheit in informationstechnischen Systemen“ [Bun13] in Deutschland, als auch durch ähnliche Dokumente in der EU und den USA, forciert wird.

Bei der Betrachtung der Sicherheit von IT-Systemen muss man zwischen „Safety“ und „Security“ eines Systems unterscheiden. Während „Safety“ die Ausfallsicherheit durch Fehler im System durch die eingesetzte Technik oder auch eventuelle Fehlbedienungen behandelt, befasst sich „Security“ mit gezielten Angriffen von Dritten auf ein System. Im Bereich von „Safety“ haben sich durch Industrie und Wissenschaft über die vergangen Jahre anerkannte Standards für die Risikoeinschätzung und Bewertung entwickelt, die auf quantitativen Verfahren basieren. So wurden beispielsweise durch IEC61508/IEC61511

die „Safety Integrity Level“ eingeführt [Com99]. Die Einführung solcher Verfahren im Kontext der „Security“ hat bislang nicht stattgefunden und eine Übertragung ist aufgrund des stark unterschiedlichen Bedrohungsmodells nicht ohne Weiteres möglich. Bei Analysen auf „Security“-Risiken kommen daher meist qualitative Verfahren zum Einsatz, die das Risiko für einen Angriff von Experten in vorgegeben Kategorien einstufen lassen. Durch diese subjektiven Einschätzungen kann es allerdings leicht zu fehlerhaften und intransparenten Bewertungen kommen, da selbst die Meinung von Experten durch einige externe Umstände beeinflusst werden kann, wie es von Kahneman und Tvertsky in [KT79] dargestellt wird. Durch diese Einflüsse kann es dazu kommen, das die Risikoanalyse, welche den Startpunkt jedes Sicherheitskonzeptes darstellt, Schwachstellen aufweist.

Die vorliegende Arbeit liefert einen ersten Ansatz für ein Verbesserung der Bewertung dieser Risiken durch eine feingranulare Bewertung einzelner Elemente in Verbindung mit einem systematischen Analyseprozess. Hierfür wird zuerst in Abschnitt 2 ein Überblick über andere Arbeiten auf diesem Gebiet gegeben und anschließend in Abschnitt 3 eine Gesamtüberblick über den systematischen Analyseprozess und seine Einbettung in die restlichen Entwicklungsschritte gegeben. In den Unterabschnitten 3.1 bis 3.4 werden die einzelnen Schritte der Analyse im Detail betrachtet. Die Ergebnisse werden abschließend in Abschnitt 4 zusammengefasst und weitere Arbeiten werden in Abschnitt 5 aufgezeigt.

2 Related Work

Auf dem Gebiet der Sicherheitsanalyse-Verfahren haben sich über die Jahre einige Standard-Verfahren etabliert und wurden durch verschiedene Institutionen standardisiert. So entstand von der „International Organization for Standardization“ die 2700x-Reihe [Int11], die sich mit Sicherheitsmanagement in der Informationstechnik befasst. Vom deutschen „Bundesamt für Sicherheit in der Informationstechnik“ (BSI) wurde das IT-Grundschutzhandbuch [Bun08b] geschaffen um Unternehmen einen Leitfaden zur Herstellung eines grundlegenden IT-Schutzes an die Hand zu geben. Die genannten Verfahren stellen dem Nutzer ein einfaches Verfahren zur Risikoanalyse zur Verfügung, welches jedoch auf einer qualitativen Schätzung des Risikos für eine Gefahr beruht. Diese ist durch den Analysten, bzw. durch einen Experten durchzuführen.

Neben dem Grundschutzhandbuch wurde vom BSI auch ein Verfahren zur generellen Analyse von kritischen Infrastrukturen geschaffen (AKIS)[Bun08a], welches eine Identifikation von möglichen Angriffspunkten innerhalb eines kritischen Infrastruktur Sektors ermöglicht. Eine Auflistung von weiteren Analyseverfahren kann ISO/IEC 31010 entnommen werden [Int09].

3 Prozess

Der in dieser Arbeit vorgestellte Prozess zur Sicherheitsanalyse zielt darauf ab, die bisherige Ermittlung von zu betrachtenden Gefahren für die Implementierung und Evaluation

von sicherheitskritischen Systemen zu verbessern und die bisher oft forcierte Nutzung von „Expertenwissen“ [Bun08b] durch einen ingenieurmäßigen Prozess zu verbessern.

Dieser Prozess besteht aus einer anfänglichen semi-statischen Analyse und einem darauf folgenden kontinuierlichen rückgekoppelten Prozess (eine Übersicht bietet Abbildung 1), der sich an den Regelkreisen aus [Int05] orientiert. Die semi-statische Analyse, welche die Analyse der Assets des Systems, möglicher Ziele eines Angreifers sowie Angreiferklassen umfasst, muss nur einmalig durchgeführt werden. Es ist allerdings zu empfehlen im Falle von technischen Neuerungen eine erneute Evaluation der Bewertungen durchzuführen, da der technische Fortschritt zu einer Änderung der Mittel, beziehungsweise deren Durchführbarkeit und Komplexität führen kann.

Aus diesem ermittelten Domänenwissen wird anschließend eine Bewertung anhand gewisser Verfahren erstellt, die im Folgenden vorgestellt werden. Abschließend wird aus dieser Bewertung und einem vorgegebenen geforderten Sicherheitslevel (Security Level - SL [Com13]) ein Report erzeugt, welcher die potentiellen zu betrachtenden Gefahren für das System angibt. Dieser Report kann für einen weiteren technischen Design Prozess genutzt werden oder als Grundlage für eine spätere Evaluation der Implementierung des Systems verwendet werden. Insofern nach dem Report neue Sicherheitsmechanismen in das System eingebunden werden, so werden diese im Rahmen des technischen Design-Prozesses umgesetzt und wirken auf die Anwendbarkeit der Mittel eines Angreifers ein und erhöhen die nötigen Ressourcen, bzw. entfernen Mittel komplett. Dieser Prozess kann in beliebig vielen Iterationen wiederholt werden. Auch bereits existierende Mechanismen können auf diese Art und Weise berücksichtigt werden.

3.1 Asset-Analyse

Als Startpunkt für die Sicherheitsanalyse sollte man sich zuerst mit der Domäne des Systems vertraut machen und analysieren, welche Rahmenbedingungen (rechtliche Vorschriften, Ziele, Abläufe, etc.) in dieser gelten. Zudem müssen Experten aus der Domäne einbezogen werden, da nur durch diese ein umfangreiches Domänenwissen, gerade in Bezug auf Erfahrungswerte, eingebracht werden kann. Wenn ein entsprechendes Wissen über das Einsatzumfeld des Systems vorhanden ist, so müssen die Assets der Domäne erfasst werden, da diese den Grundstein für die folgende Analyse darstellen. Erst wenn man wirklich verstanden hat, was im zu schützenden System wichtig ist, dann bekommt man eine Vorstellung davon, was ein Gegner angreifen könnte.

In den meisten kritischen Infrastrukturen können die Assets in drei Kerngruppen unterteilt werden: Betriebliche Assets (z.B. Sicherheit des Betriebs), technische Assets (z.B. Intakte Infrastruktur) und Wahrnehmungs-Assets (wie das Image des Unternehmens).

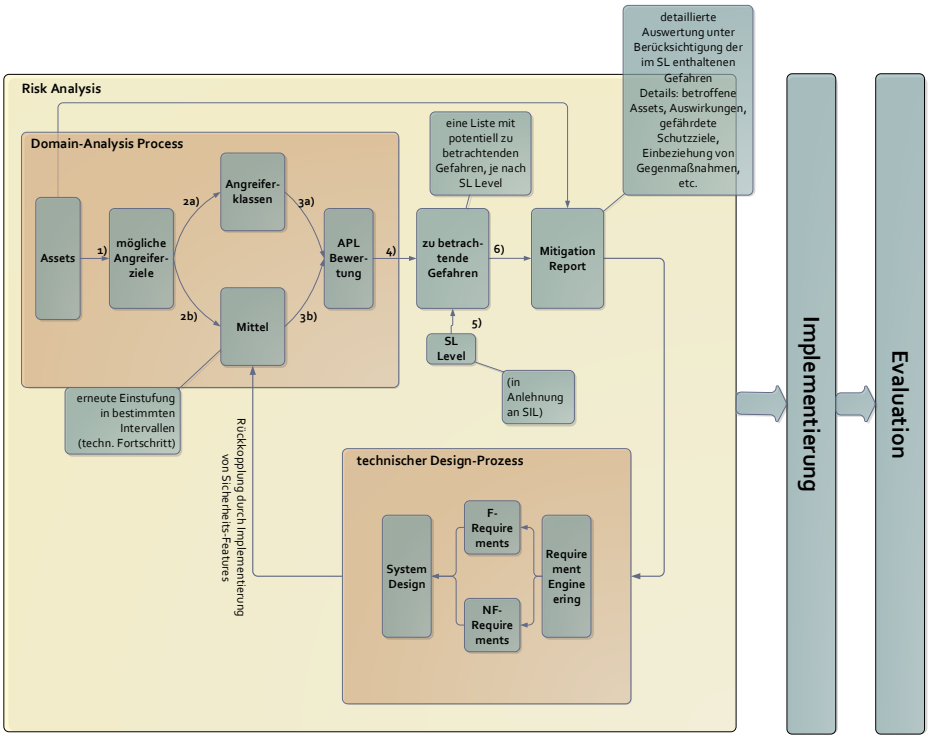


Abbildung 1: Analyseprozess im Kontext

3.2 Ziele von Angreifern

Anhand der Assets des zu betrachtenden Systems werden nun mögliche Ziele, welche ein Angreifer verfolgen könnte, abgeleitet. Hierfür führen wir anhand der Ergebnisse aus der vorausgegangenen Analyse der Domänen-Assets nun eine Identifikation von potentiellen Angreiferzielen durch. Zur Sammlung von möglichen Kandidaten bietet es sich an auch auf andere (ähnliche) Domänen zurückzugreifen und in einem folgenden Schritt zu bewerten, ob das mögliche Ziel aus dieser Domäne auch in der aktuell betrachteten Domäne realistisch ist oder nicht.

Das angewendete Vorgehen zur Identifikation von Angreiferzielen ist in Abbildung 2 veranschaulicht. Zu Beginn erfolgt die Auffüllung eines ungewerteten Ziele-Pools. Hierfür empfiehlt es sich, zur ersten Identifikation von möglichen Zielen, Experten mit einem Wissen aus der Domäne einzubeziehen und mit diesen ein gezieltes Brainstorming durchzuführen. Bei diesem Brainstorming sollte auf den bereits identifizierten Assets des Systems aufgebaut werden und zusammen mit bereits erfolgten Angriffen oder Gefährdungen auf ein Vorgängersystem ein gewisser Grundpool an Angreiferzielen gesammelt werden

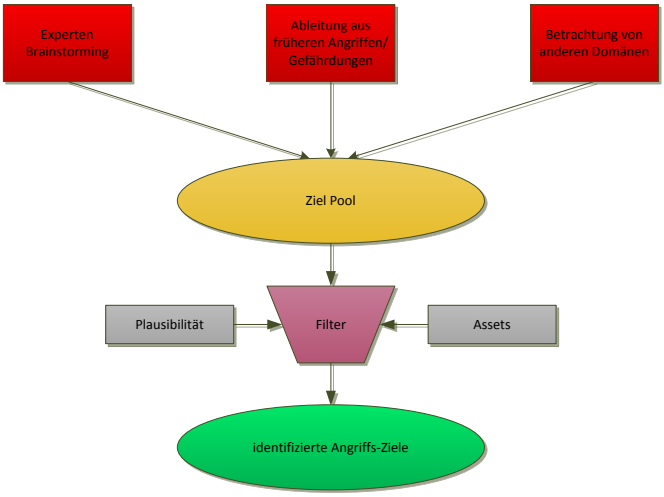


Abbildung 2: Identifikation der Angreifer-Ziele

können. Anschließend empfiehlt es sich auch einen Blick über die eigene Domäne hinaus zu werfen und zu betrachten, welche Angriffe auf andere ähnliche Sicherheitssysteme erfolgt sind und die daraus neu entstandenen Ziele mit in den Pool aufzunehmen.

Abschließend muss der Pool an identifizierten Angreiferzielen auf Plausibilität geprüft werden. Außerdem sollte geprüft werden, welches der Ziele zu welchem der Assets korrespondiert. Sollte ein Ziel zu keinem der Assets korrespondieren, so ist dieses noch einmal gezielt auf seine Plausibilität zu prüfen. Ist das Ziel plausibel, so sollte überlegt werden, ob eventuell ein Asset übersehen wurde.

3.3 Klassifikation

Nach der Identifikation der Ziele lassen sich anhand der identifizierten Ziele und Assets in einem anschließenden Schritt nun potentielle Angreiferklassen identifizieren und nach verschiedenen Charakteristika bewerten. Die Identifikation findet im Allgemeinen durch ein Brainstorming von Verantwortlichen und Experten statt und sollte klar umrissene Angreiferklassen liefern.

Generell lässt sich bei Angreifern zwischen drei Gruppen unterscheiden:

Gezielte Angreifer
Angreifer, die das aktuelle System mit einem bestimmten Ziel angreifen und sich

der Konsequenzen ihres Handelns bewusst sind (z.B. ein Hacker, der eine Firewall knackt).

Versehentliche Angreifer
Angreifer, die das System eher aus Zufall, bzw. unbeabsichtigt schädigen, z.B. ein Computer-Wurm, der eigentlich auf ein bestimmtes Ziel gerichtet war, allerdings durch eine gewisse Ähnlichkeit des zu schützenden Systems auch dieses angreift (Prominentes Beispiel: Stuxnet, dessen eigentliches Ziel Zentrifugen waren und der danach auf zahlreiche Industrieunternehmen übersprang).

Nicht-menschliche Gefahren
Hierbei handelt es sich um Gefahren, die nicht zwingend durch eine gezielte menschliche Handlung hervorgerufen werden (z.B. eine Überschwemmung oder ein Kabelbruch durch Materialermüdung). Diese werden primär durch Methoden der „Safety“ behandelt.

Im Anschluss an die Identifikation findet die Klassifikation nach gewissen Charakteristika statt (im weiteren Verlauf betrachten wir nur die Gruppe der „Gezielten Angreifer“). Für diese Einstufung werden die folgenden Charakteristika genutzt und abschließend entsprechend ihrer Ausprägung gewertet:

Motivation:
1: Ein Angriff wird nur unter perfekten Bedingungen durchgeführt
5: Ein Angriff wird trotz aller Widrigkeiten stattfinden

Vorhandenes Expertenwissen, bzw. Zugriff auf Expertenwissen:
1: Expertenwissen ist für den Angreifer (nahezu) unzugänglich
5: Expertenwissen ist bei dem Angreifer bereits vorhanden, bzw. unmittelbar zugänglich

Vorhandenes Insiderwissen, bzw. Zugriff auf Insiderwissen:
1: Insiderwissen ist für den Angreifer (nahezu) unzugänglich
5: Insiderwissen ist bei dem Angreifer bereits vorhanden, bzw. unmittelbar zugänglich

Zugriff auf Spezialgeräte:
1: Sind durch den Angreifer nicht zu beschaffen
5: Sind bereits beim Angreifer vorhanden

Zugriff auf finanzielle Ressourcen:
1: Keine oder nur sehr geringe finanzielle Mittel vorhanden
5: Dem Angreifer stehen nahezu unbegrenzte Mittel zur Verfügung

Vorbereitungszeit:
1: Keine, es handelt sich um einen spontanen Täter
5: Sehr lange Zeit für eine intensive Vorbereitung

Ausgangsbasis:
1: Der Angreifer agiert vor Ort und es besteht ständig das Risiko entdeckt zu werden
5: Der Angreifer kann sich entfernt vom Ziel vorbereiten und Tests durchführen

Entdeckbarkeit des Angreifers:

1: Eine Entdeckung hätte für den Angreifer untragbare Konsequenzen
5: Eine Entdeckung wäre für den Angreifer sogar noch ein Bonus (z.B. bei terroristischen Anschlägen)

Personelle Ressourcen:

1: Es handelt sich um einen Einzeltäter
5: Es steht ein reichlicher Fundus an Personal zur Verfügung

Die einzelnen Charakteristika werden durch ein Punktesystem gewertet, welches von 1 bis 5 reicht, wobei verallgemeinert gilt: 1 = ungefährlich, da keine Ressourcen in Hinblick auf diese Charakteristik vorhanden sind; 5 = gefährlich, es sind große Ressourcen für diese Charakteristik vorhanden. Zwischen diesen Werten erfolgen jeweilige Abstufungen. Ein exemplarisches Beispiel aus der Bewertung kann Abbildung 3 entnommen werden.

	Motivation	Expertenwissen	Insiderwissen	Spezialequipment	Geld	Vorbereitungszeit	Ausgangsbasis	Entdeckbarkeit	Personal	Angreiferpotential
Vandalen	2	1	1	1	1	1	1	2	1	1,22
Hacker	3	5	2	3	2	4	3	1	1	2,67
Computerkriminelle	4	5	3	4	3	4	3	2	2	3,33
Terroristische Organisationen	5	2	2	2	4	5	5	5	3	3,67
Andere Staaten (Konflikt)	4	5	5	5	5	5	5	4	5	4,78

Abbildung 3: Auszug aus der Bewertung potentieller Angreiferklassen (Domäne: Leit- und Sicherungstechnik der Eisenbahn)

Zur Gesamtbewertung der Angreiferklasse wird noch ein abschließender Score eingeführt: das Angreiferpotential. Das Angreiferpotential ist eine einfache Abschätzung der Gesamtgefahr, die von einer bestimmten Angreifergruppe an sich ausgeht. Es wird später im Rahmen der Bewertung in Zusammenhang mit dem SL zur Selektion der relevanten Angreifer benötigt. Es berechnet sich wie folgt:

$$\begin{aligned} \text{Menge der Charakteristika } & C \\ \text{Bewertung der Charakteristik: } \forall c \in C : & R_c = \{1, .., 5\} \\ \text{Angreiferpotential } & P_{Angreifer} = \frac{\sum R_c}{|C|} \end{aligned} \tag{1}$$

Aus dem beschriebenen Vorgehen ergibt sich letztendlich die Bewertung für unsere betrachteten Angreifer und diese werden nach ihrem Potential gelistet. Einen ähnlichen Prozess durchlaufen auch die zur Realisierung der Ziele denkbaren Mittel der Angreifer. Diese werden analog zu den Angreifern klassifiziert, allerdings werden diese nach dem entsprechenden Bedarf an Ressourcen eingestuft.

3.4 Bewertung

In einer weiteren Phase werden nun die separat voneinander bewerteten Angreiferklassen und Angreifermittel zusammengeführt. Hierdurch wird ermittelt, auf welche Schwachstellen man sich besonders konzentrieren sollte, da deren Eintreten als wahrscheinlich anzusehen ist.

Für eine bessere Entscheidbarkeit beim gezielten Absichern gegen Schwachstellen werden wir die einzelnen Mengen nun in einem Mapping-Prozess kombinieren und auf diese Weise eine Bewertung für die jeweilige Kombination ermitteln. In einem dem Mapping-Prozess vorgelagerten Schritt wird die Angreifer-Liste gemäß dem gewählten SL-Level gefiltert. Hierdurch wird der aufwendige Mapping-Prozess nur für relevante Angreifer-Mittel Kombinationen durchgeführt.

Um diese Auswahl für die zu betrachtenden Gefahren für den Security Level des untersuchten Systems zu treffen, wurde zur Vereinheitlichung, in Anlehnung an [Com99] und [Com13], der SL eingeführt, der die Stufen 1 bis 4 umfasst. Je höher der SL, umso mehr Angreifer und damit auch Gefährdungen werden in der Liste der zu betrachtenden Gefahren berücksichtigt.

$$\text{SL1} \subseteq \text{SL2} \subseteq \text{SL3} \subseteq \text{SL4} \tag{2}$$

Als erste echte Mapping Phase werden nun die identifizierten Schwachstellen und Mittel M auf die bereits zuvor identifizierten und gefilterten Angreifer A über die Charakteristika C gemappt. Aus dem Mapping-Prozess resultiert das Gefahrenpotential P für diese spezielle Angreifer-Mittel Kombination.

Im Rahmen des Mapping-Prozesses betrachten wir die vorher ermittelten Werte der Angreifer und der Mittel als Ressourcen, bzw. Ressourcenkosten. Daher ergeben sich die folgenden Relationen:

$$\text{Expertenwissen} \leftrightarrow \text{benötigtes Expertenwissen} \tag{3}$$
$$\text{Insiderwissen} \leftrightarrow \text{benötigtes Insiderwissen} \tag{4}$$
$$\text{Spezialequipment} \leftrightarrow \text{benötigtes Spezialequipment} \tag{5}$$
$$\text{Geld} \leftrightarrow \text{Kosten} \tag{6}$$
$$\text{Vorbereitungszeit} \leftrightarrow \text{benötigte Vorbereitungszeit} \tag{7}$$
$$\text{Entdeckbarkeit} \leftrightarrow \text{Entdeckbarkeit des Angreifers} \tag{8}$$
$$\text{Personal} \leftrightarrow \text{benötigtes Personal} \tag{9}$$

Anschließend lässt sich noch ein Gesamtscore für die jeweilige Gefahr errechnen. Dieser Gesamtscore berücksichtigt bei der Berechnung speziell auch noch die Motivation des Angreifers, denn diese hat eine starke Aussagekraft darüber, unter welchen Bedingungen ein Angriff durchgeführt werden würde. Wodurch wir die folgende Berechnung für den Score erhalten:

$$\text{Score}(A_m) = \begin{cases} \frac{\text{Angreifer}_{\text{Mittel}} \times \text{Angreifer}_{\text{Motivation}}}{5} & \text{if Angreifer}_{\text{Mittel}} \geq 0 \\ \varepsilon & \text{if Angreifer}_{\text{Mittel}} < 0 \end{cases} \tag{10}$$

Zuletzt lässt sich anhand dieser Scores ein Gefahren-Report erstellen, der während der Implementierung oder während einer weiteren Design Phase verwendet werden kann um gezielt Maßnahmen gegen bestimmte Bedrohungen zu implementieren. Ein Ausschnitt aus einem exemplarischen Report ist in Abbildung 4 zu sehen.

In der „Liste der noch zu behandelnden Gefahren“ des Reports wird eine Auflistung der möglichen Gefahren aufgezeigt. In dieser sind jeder Gefahr die betroffenen Assets zugeordnet, sowie die Menge der Angreiferklassen, durch welche die Gefahr genutzt werden könnte. Unter dem Punkt „Größte Gefahr“ wird die schwerwiegendste Angreiferklasse mit deren Potential dargestellt. Die Gefahren selbst sind nach deren Gefahrenscore sortiert. Jede Gefahr aus dieser Liste lässt sich unter dem Punkt „Gefahr-Details“ im Detail anzeigen. Hierbei werden die jeweils erforderlichen Mittel zur Ausnutzung der Gefahr aufgelistet und die zur Nutzung dieses Mittels fähigen Angreifer aufgeführt. Insofern ein Mittel eine Komposition aus verschiedenen Mitteln ist, kann jedes dieser Mittel in seine Komponenten aufgelöst werden. Dies kann unter dem Punkt „Bedingungen“ gesehen werden. Hierbei sind sowohl $\wedge$ als auch $\vee$ Verknüpfungen möglich. Zur anschaulichen Visualisierung kann man die Beziehung von Gefahren und deren Mittel auch in Form eines Angriffs-Baums, wie in [Sch99] beschrieben darstellen. Abschließend kann man sich zu jedem Mittel die möglichen Angreiferklassen anzeigen lassen, hierbei sind sämtliche mögliche Klassen in einer Tabelle nach deren Angreifer-Potentialen geordnet (siehe Abbildung 4 unten).

4 Zusammenfassung

In der vorliegenden Arbeit wurde ein neuer Prozess zur Analyse der Sicherheitsanforderungen an ein sicherheitskritisches IT-System vorgestellt. Es wurde gezeigt, wie durch ein methodisches Vorgehen Assets und Ziele von Angreifern identifiziert werden können und wie diese, sowie deren Mittel selbst, gezielt analysiert und bewertet werden können. Außerdem wurde dargestellt, wie mittels einer feingranularen Schätzung über die Fähigkeiten der Angreifer und die Anforderungen von möglichen Angreifern ein Bild der existierenden Gefährdungen für das System modelliert werden kann. Im Rahmen dieser Modellierung wurde ein Verfahren zur Durchführung der Risikoanalyse im Kontext des SL vorgestellt, durch diesen SL wird ein ähnlicher Indikator, wie der SIL für die Safety eines Systems, auf Security übertragen. Abschließend wurde eine übersichtliche Darstellung in Form eines Reports vorgestellt.

Durch den hier vorgestellten Prozess erfolgt eine deutliche Steigerung der Transparenz bei der Analyse der Security Anforderungen und die Arbeit des Analysten wird weniger fehleranfällig, da er nur kleine Fragmente schätzen muss und nicht eine große Gesamtwertung. Das generelle Vorgehen ist an den „risk management process“ der ISO 27005 angelehnt. So wird in den ersten Schritten ein Kontext geschaffen, danach die Risiken identifiziert und abschließend bewertet. Die Behandlung der Risiken selbst müsste anschließend

Liste der noch zu behandelnden Gefahren

Gefahr	Betriebl. Auswirkung	Betroffene Assets	Zahl der Angreiferklassen	Größte Gefahr	Score
Modifikation der Infrastruktur (6.2.17)	b,g	(5.1.2), (5.2.1), (5.2.3), (5.2.4), (5.3.1)	4	kk – 0.6	0.6
...	...	...	...	...	...

Gefahr-Details (6.2.17)

Mittel	Nutzbar durch	Entdeckbarkeit	Mittel-Score
Extraktion von krypt. Keys (8.7.1)	ko, kk, fm, ck	Sehr gering	0.6

Mittel-Details (8.7.1) - Bedingungen

Erforderliches Vorgeschehen	Nutzbar durch	Entdeckbarkeit	Mittel-Score
DD oder MdK (8.7.6 oder 8.9.1)	to, ko, kk, fm, ha, ck, er	Sehr gering	0.78
EkS (8.7.1)	to, ko, kk, fm, ha, ck, er	Sehr gering	0.54

Mittel-Details (8.7.1) – Angreiferklassen (EkS)

Angreifer	Angreifer-Potential
ck	0.914
...	...

Abbildung 4: Auszug aus dem Gefahren Report

durch eine Änderung des Designs erfolgen, was allerdings nicht zum Umfang dieser Arbeit gehört. Der Unterschied zu den bisher existierenden Arbeiten liegt in der Feingranularität der Schätzungen: Während sich aktuell oft mit einer einfach Schätzung für die Eintrittswahrscheinlichkeit beholfen wird, wird diese Schätzung hier in einzelne Charakteristika von Angreifer und Mittel zerlegt, wodurch der mögliche Fehler minimiert wird. Die anschließende Erstellung eines Scores für die Schwere des Problems, das durch diese Kombination entsteht, beruht auf einfachen nachvollziehbaren Algorithmen. Die Entscheidung über die Relevanz der Risiken erfolgt zudem über wohl definierte SL-Werte.

Auch wenn im Zusammenhang mit dem Prozess noch einige weitere Punkte betrachtet werden müssen und es sich bei der hier vorgestellten Variante erst um ein erstes Konzept handelt, so sollte die Steigerung an Genauigkeit, das methodischere Vorgehen und die Minimierung möglicher Fehlerquellen bereits gut erkennbar sein.

5 Zukünftige Arbeiten

Die weiteren Arbeiten auf dem Gebiet dieser Arbeit sind vielfältig: Zum einen wird eine Methodik geschaffen werden, welche es ermöglicht Angreifer nicht nur als einzelne Individuen, sondern auch als kooperierende Subjekte zu betrachten und abzubilden. Außerdem wird zur Steigerung der Aussagekraft in Hinblick auf das Schadensausmaß einer Gefahr eine Bewertung der Assets eingeführt werden, welche eine automatisierte Schätzung des möglichen Schadens auf ein System durch eine Gefahr ermöglicht. Um die spätere Arbeit der Analysten weiter zu erleichtern und die Analysen wirtschaftlicher zu machen wird des weiteren versucht eine Möglichkeit zu schaffen aus dem Report direkt ein „Protection Profile“ nach „Common Criteria“ abzuleiten oder zumindest Dokumente zu liefern, welche bei der Erstellung eine gute Hilfestellung geben.

Literatur

[Bun08a] Bundesamt für Sicherheit in der Informationstechnik. *Analyse Kritischer Infrastrukturen - Die Methode AKIS*. Bundesamt für Sicherheit in der Informationstechnik, 2008.

[Bun08b] Bundesamt für Sicherheit in der Informationstechnik. *Risikoanalyse auf der Basis von IT-Grundschutz (Standard 100-3)*. Bundesamt für Sicherheit in der Informationstechnik, 2008.

[Bun13] Bundeministerium des Innern. *Entwurf eines Gesetzes zur Erhöhung der Sicherheit informationstechnischer Systeme*. Bundeministerium des Innern, März 2013.

[Com99] International Electrotechnical Commission. *Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/elektrisch programmierbarer Systeme (IEC 61508)*. International Electrotechnical Commission, 1999.

[Com13] International Electrotechnical Commission. Industrial communication networks - Network and system security (IEC 62443). *International Electrotechnical Commission*, 2013.

[Int05] International Organization for Standardization. *Information security management systems - Requirements (ISO/IEC 27001:2005)*. International Organization for Standardization, 2005.

[Int09] International Organization for Standardization. Risk Management - Risk assessment techniques (ISO/IEC 31010:2009). *International Organization for Standardization*, 2009.

[Int11] International Organization for Standardization. *Information security risk management (ISO/IEC 27005:2011)*. International Organization for Standardization, 2011.

[KT79] Daniel Kahneman und Amos Tversky. Prospect Theory: An Analysis of Decision under Risk. *ECONOMETRICA*, 47(2):263–292, March 1979.

[Sch99] Bruce Schneier. Attack Trees - Modeling security threats. *Dr. Dobb's Journal*, December 1999.

Bestimmung des technical-Value at Risk mittels der bedingten Wahrscheinlichkeit, Angriffsbäumen und einer Risikofunktion

Wolfgang Boehmer

wboehmer@cdc.informatik.tu-darmstadt.de
Technische Universität Darmstadt, Morneweg Str. 30, CASED building,
64293 Darmstadt, Germany

Abstract: In diesem Artikel wird ein Vorschlag zur Berechnung eines technical-Value at Risk (t-VaR) diskutiert. Dieser Vorschlag basiert auf der Risikoszenarientechnik und verwendet die bedingte Wahrscheinlichkeit gemäß Bayes. Zur Ermittlung der Bedrohungen werden Angriffsbäume und Angriffsprofile eingesetzt. Die Schwachstellen sind empirisch für eine Versicherung im Jahr 2012 ermittelt worden. Die Angriffsbäume werden gewichtet mit einer Risikofunktion, die die kriminelle Energie beinhaltet. Zur Verifizierung dieser Vorgehensweise ist die VoIP-Telefonie einer Versicherung der t-VaR berechnet worden. Es zeigt sich, dass dieses Verfahren hinreichend gute Ergebnisse erzielt und für den technischen Bereich eine wirkungsvolle Methode darstellt[1].

Bedingte Wahrscheinlichkeit; Bayes Theorem; Angriffsbäume; Bedrohungsprofile; Risikofunktion; Risikoszenarien.

1 Einleitung

In den Anfangszeiten der Absicherung der IT, also in den frühen 90-er Jahren, lag der Schwerpunkt rein auf der technischen Absicherung. Denn man hatte zu der Zeit erkannt, dass die IT verwundbar ist. Diese oftmals kostenintensiven Maßnahmen waren zu der Zeit nicht immer an betriebswirtschaftliche Überlegungen gebunden und ein Bezug zu den Geschäftsprozessen wurde nur rudimentär hergestellt. Einen Meilenstein stellt daher der BASEL II Accord im Jahre 2004 dar, da dieser die operationellen Risiken gleichberechtigt neben den Markt und Finanzrisiken stellte. Im gleichen Zeitraum hat sich bei den Markt- und Finanzrisiken eine Bemessungsgröße für die Risiken durchgesetzt, die mit Value at Risk (VaR), respektive seine kohärente Variante der C-VaR, bezeichnet wird. Ferner wurden für die operationellen Risiken in etlichen theoretischen Modellen versucht entsprechende Bemessungsgrößen zu dem VaR zu ermitteln, die sich in der Praxis jedoch nie durchgesetzt haben. In Folge dessen entstanden die qualitativen und die quantitativen Risikoverfahren, die bis heute noch nebeneinander stehen.

[1] Dieser Beitrag ist in abgewandelter Form in englischer Sprache auf der ARES 2013 Konferenz in Regensburg (`http://www.ares-conference.eu/ares2013/`) vorgestellt und ist im Tagungsband abgedruckt.

Eine wesentliche Besonderheit in der Berechnung der Risiken zwischen den Markt- und Finanzrisiken einerseits und den operationellen Risiken andererseits liegt nun darin, dass im Bereich der operationellen Risiken die IT-Infrastruktur durch Bedrohungen und vorhandenen Schwachstellen geprägt ist[2]. Somit führt der bei den Markt- und Finanzrisiken weit verbreitete Ansatz, eine Monte Carlo Simulation zur Ermittlung der Wahrscheinlichkeitsverteilung zu verwenden, bei den operationellen Risiken zu keinen befriedigenden Ergebnissen.

Generell basiert die Risikoanalyse auf die Betrachtung zweier Verteilungen. Die erste Verteilung beschreibt die Eintrittswahrscheinlichkeit von Risikoereignissen und die zweite Verteilung stellt die Auswirkung des Risikoereignisses dar, wenn das Risiko eingetreten ist. Anschließend werden die beiden Verteilungen gefaltet. Diese gefaltete Verteilung ist eine neue Verteilung, die Risikoverteilung. Wird von dieser z.B. ein 5%-Quantil als erwarteter Verlust definiert, so ist der VaR bestimmt. Allein die Frage bleibt, wie werden realitätsnahe Verteilungen gefunden. Die Type der Eingangsverteilung bestimmt einen spezifischen Teil im Bereich des opRisk. Wir gehen im Anhang weiter auf diese Zusammenhänge ein.

Die Forschungsfrage besteht nun darin eine Methode zu entwickeln mit der ein VaR für den technischen Bereich möglich ist (t-VaR) und die Besonderheiten einer IT-Infrastruktur berücksichtigt, jedoch im Ergebnis mit den VaR der Markt- und Finanzrisiken gleich zu setzen ist.

Der restliche Artikel ist wie folgt organisiert: Im Anhang wird das zu Grunde liegende Modell diskutiert. Im Abschnitt 2 wird die Datenerhebung zur Ermittlung der Schwachstellen, die Bedrohung und die Anwendung des Models zur Berechnung des t-VaR am Beispiel einer Versicherung für deren VoIP-Anlage diskutiert. Anschließend folgt im Abschnitt 3 die Diskussion der Ergebnisse. Im Anhang und dort im Abschnitt 5.2 wird die relevante Literatur diskutiert. Der Artikel schließt mit einer kurzen Zusammenfassung, weiterführenden Überlegungen und Vorschlägen zu weiteren Untersuchungen ab.

2 Data acquisition

Gemäß dem Model, beschrieben im Anhang, wird in diesem Abschnitt die Datenerhebung bzgl. der Schwachstellen vorgenommen und, darauf abgestimmt, der attack-tree mit den dazugehörigen threat agent (actor) analysiert, um Risikoszenarien gemäß der Gleichungen 5.9 - 5.12 zu entwickeln. Dabei repräsentiert ψ die VoIP-Telephony. Andere Systeme werden nicht untersucht und so ist $\Psi = \psi$. Dies bedeutet, dass sich alle Schwachstellen, Bedrohungen und Risikoszenarien auf ψ = VoIP-Telefonie beziehen.

In der Abbildung 1 ist die VoIP-Telefonie der Versicherungsfirma grob skizziert. Es handelt sich um zwei Lokationen A, B. Die beiden Lokationen (A, B) sind über ein nicht öffentliches Netz (MPLS Cloud) verbunden und an dem Switch an der Lokation A und B werden sowohl der Datenverkehr als auch der Sprachverkehr abgehandelt. Die MPLS

[2] Anzumerken ist, dass die operationellen Risiken mehr beinhalten als nur die IT-Infrastruktur gemäß Basel II.

Cloud wird von einem Service Provider unterhalten, der neben diesen Kunden auch andere Kunden in seiner MPLS Cloud versorgt. Die Kunden des Versicherungsunternehmens wählen von außen die softphones der Mitarbeiter der Versicherung an. Neben der Erreichbarkeit (Verfügbarkeit) ist ebenso die Vertraulichkeit der Gespräche ein Sicherheitsziel für die Versicherung.

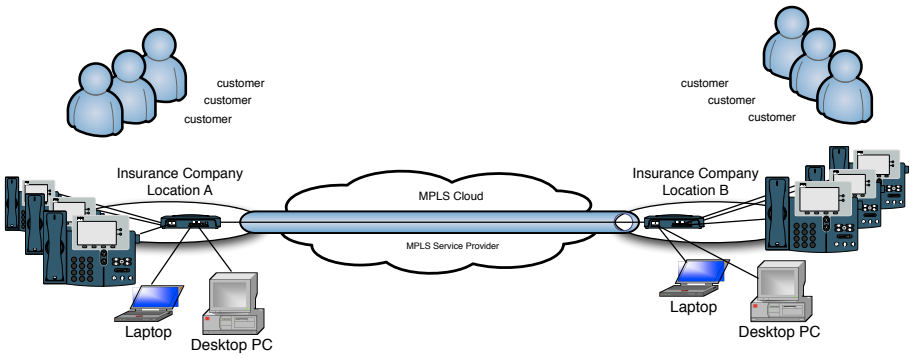


Abbildung 1: VoIP Architektur der Versicherung

2.1 Analyse der Schwachstellen

Die empirische Untersuchung in 2012 für die Versicherungsfirma hatte zum Ziel die gegenwärtigen Schwachstellen in dem Scope der VoIP-Telefonie zu analysieren. In diesem Abschnitt werden Schwachstellen exemplarisch diskutiert, um die Vorgehensweise des Models (vgl. Anhang) zu zeigen.

Die Analyse wurde in zwei Schritten durchgeführt. Im ersten Schritt wurde das VoIP-Modul (B. 4.7) der IT-Grundschutzkataloge des BSI verwendet [BSI08, mOfSiIT05]. Es wurden mittels Interview die technische Abteilung befragt. Im zweiten Schritt wurde ein *white box pentest* durchgeführt, um die Aussagen zu verifizieren und um weitere Schwachstellen zu finden. Erwähnenswert ist, dass es zwischen der VoIP-Telephony und dem Internet keine Verbindung existiert.

In der Untersuchung wurde unter anderem erkannt, dass die Sprachdaten (RTP- Stream) nicht verschlüsselt übertragen werden und die Ports an den *Softphones* und die *Patchdosen* in den Räumen nicht gegen unerlaubte Nutzung gesichert waren (IEEE 802.1x, Port Based Network Control, PBNC). Weiterhin wurde erkannt, dass keine Firewall zwischen dem MPLS Netz und dem LAN der Insurance Company existierte ebenso wurden die Sprachdaten in dem MPLS-Netz unverschlüsselt übertragen.

Basierend auf diesen Schwachstellen wurde im Anschluß eine Bedrohungsanalyse durchgeführt.

2.2 Analyse der Bedrohungen

Die Bedrohungsanalyse basiert auf den identifizierten Schwachstellen und beinhaltet drei Elemente, die Bedrohungsszenarien, die dazu passenden Bedrohungsprofile und die jeweilige Funktion der kriminellen Energie.

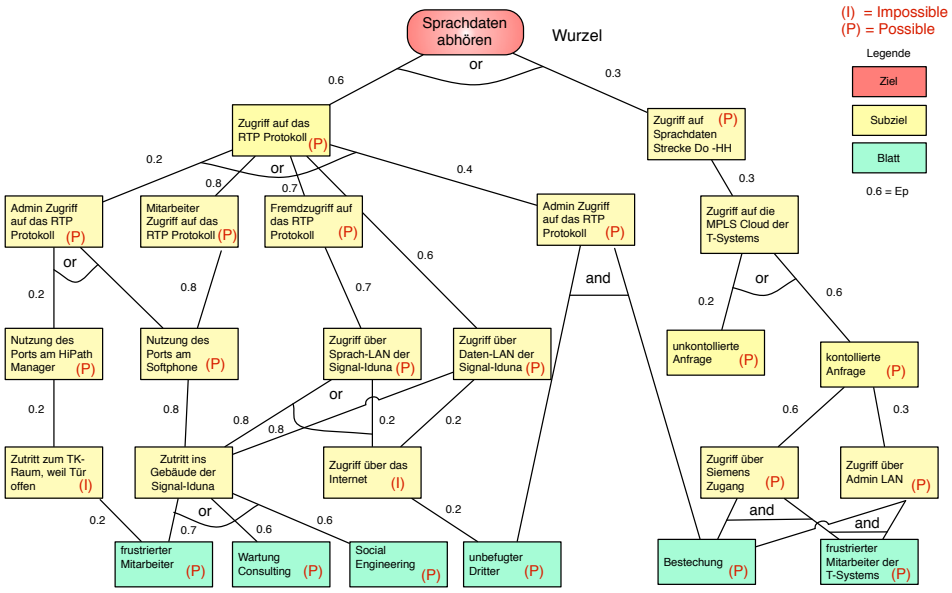


Abbildung 2: Angriffsbaum mit Werten der kriminellen Energie

In der Bedrohungsanalyse zeigen wir exemplarisch wie für das Schutzziel Vertraulichkeit der VoIP-Telefonie die aktuelle Bedrohungssituation für die Versicherungsfirma gegeben ist. Vertraulichkeit für die VoIP-Telefonie bedeutet, dass die Gespräche zwischen den Mitarbeitern der Versicherungsfirma mit ihren Versicherten nicht abgehört werden dürfen. Ebenso müssen die Gespräche zwischen den Standorten A und B vertraulich behandelt werden. Denn durch das Abhören könnte ein Schaden für die Versicherungsfirma entstehen.

Wie groß die Bedrohung tatsächlich und welches Bedrohungsprofil passend ist für die Versicherungsfirma wird in diesem Abschnitt diskutiert.

Die Abbildung 2 zeigt mögliche Bedrohungspfade für die Versicherungsfirma vom Blatt bis zur Wurzel. Zwischen den Pfaden sind entweder *oder* oder *und* Verbindungen. Die *und* Verbindung besagt, dass beide Pfade, die mit dem *und* verbunden sind, erfüllt sein müssen, um das nächste Teilziel zu erreichen. Zwischen den Teilzielen ändert sich die Funktion der kriminellen Energie. Die einzelnen Blätter stellen die Akteure dar. Die Wurzel stellt das Ziel des Angriffs (Abhören des Sprachverkehrs) dar. Die Ziffern stellen die Funktion der kriminellen Energie dar, die nach dem Muster der Gleichung 5.13 - 5.15 ermittelt wurden.

Die Indizes (I) = Impossible und (P) = Possible unterscheidet mögliche Pfade die aufgrund von Policies oder baulichen Gegebenheiten möglich oder nicht möglich sind.

Wie groß das Risiko des Abhörens (eavesdropping) tatsächlich ist wurde mit Risikoszenarien (vgl. Gleichungen 5.9 - 5.12) gemäß der Bayesian Statistik, den Angriffsbäumen, den *threat agent* und der *crimal function* abgeschätzt. Dabei wurde als *threat agent* ein Innentäter postuliert, weil sich der unverschlüsselte *voice traffic* nur in dem LAN der Versicherungsfirma bewegt und ebenfalls unverschlüsselt in der MPLS Cloud des Providers transportiert wird. Denn es ist keine Internet Verbindung für den Voice Traffic der Versicherungsfirma vorhanden und die MPLS Cloud ist ebenfalls nicht mit dem Internet verbunden. Somit kommt als Innentäter nur jemand aus der Versicherung oder jemand aus dem Umfeld (Administrator) des MPLS Cloud Providers in Frage.

Als Bedrohung wurde das Abhören (thr_1) von sensiblen Sprachdaten der Versicherungsfirma angenommen, die von einem Innentäter (threat agent) vorgenommen werden könnten. Ebenso könnte der Innentäter auch durch Service Personal (thr_2) oder als Eindringling (thr_3) dargestellt werden. Zu jedem Bedrohungsbaum und Innentäter wird eine eigene *criminal function* bzw. Risikofunktion aus den Abb. 5, Abb. 6 und Abb. 7 mit der Gleichung 5.16 abgeschätzt. Die Blätter in der Abb. 2 stellen die unterschiedlichen Bedrohungsprofile und Akteure dar. Über das Verhalten von Innentätern sind bereits eine Reihe von Artikeln erschienen, stellvertretend ist hier der Artikel von I. J. Martinez-Moyano et al. [MMRC+08] zitiert. Es wird mit ψ_1 die VoIP-Infrastruktur bezeichnet und die Schwachstelle der nicht verschlüsselten Sprachdaten mit Vul_1 bezeichnet.

Der möglich entstehende Schaden für die Versicherung durch den Innentäter resultiert durch das Bekanntwerden der abgehörten Sprachdaten in der Öffentlichkeit und dem damit verbundenen Image Schaden. D.h. für das Szenario Sprachdaten abhören wird eine Annahme vorgenommen, dass 25 Versicherte ihren Vertrag im laufenden Jahr kündigen. Der Verlust durch die Vertragskündigung wird mit l_{25c} dargestellt. Der Verlust von 25 Versicherungsverträgen in einem Zeitraum von einem Jahr liegt auch deutlich oberhalb der normalen Fluktuation (Fluktuationsverlust), die mit 11 Verträgen pro Jahr (T) angegeben wurde und somit deutlich oberhalb des Erwartungswertes. Die Zahl der Verluste mit 25 Verträgen entspricht ca. einem Faktor zwei des Fluktuationsverlustes mit 11 Verträgen. Der Faktor 2 - 3 ist aus ähnlichen Reputationsverlusten (ADAC[3]) abgeschätzt.

Wir können mit den Gleichungen 5.9 - 5.12 wie folgt für die Bedrohungsprofile bzw. Akteure und Angriffsbäume dieser empirischen Fallstudie (ψ = VoIP Telefonie der Versicherung) die Risikoszenarien mit dem möglichen Verlust von 25 Verträgen unter der Berücksichtigung der Gl. 5.16 ausdrücken.

Neben dem Schutzziel der Vertraulichkeit und den Risikoszenarien wie unter $R_{sz1} - R_{szn}$ (5.9 - 5.12) beschrieben, wurden auch Risikoszenarien für die anderen beiden Schutzziele, der Verfügbarkeit und der Integrität vorgenommen. Somit wurden für das Schutzziel Verfügbarkeit und Integrität analoge Bedrohungsbäume mit passenden Akteuren entwickelt, um dann die bedingte Wahrscheinlichkeit und somit die Risikoszenarien bestimmen zu können.

[3] zum Jahreswechsel 2013/2014 sind ebenfalls Vereinsmitglieder mit einem Faktor zwei (56000 Mitglieder) gegenüber der regulären Jahreskündigung (15000)

3 Ergebnisse

In diesem Abschnitt werden die Ergebnisse der Untersuchung zur VoIP-Telefonie diskutiert.

Der Geschäftserfolg einer Versicherung lässt sich u.a. an der Menge der Versicherungsverträge ($|C|$) bestimmen. Dabei schwankt die Zahl der Versicherungsverträge ($c_i \in C$) durch natürliche Fluktuation durch Kündigungen zwischen 8 und 11 Versicherungsverträge für die empirische Fallstudie. Die Zielerwartung ist pro Jahr nicht mehr als 8 Versicherungsverträge zu verlieren, jedoch ist der Verlust von 11 Versicherungsverträgen noch akzeptabel und bildet den Wert α. Damit liegt der Erwartungswert $E_w^T(C)$ für ein Jahr bei 8 Versicherungsverträgen und entspricht dem Zielwert 1 in der Abbildung 3. Der Betrachtungszeitraum beträgt ein Jahr (T).

Die Gleichung 3.1 beschreibt die negative Schwankung um den Zielwert von 1 ($l_{c8} = 8$ verlorene Verträge). Es werden die pro Jahr hinzugekommenen Verträge in unserer empirischen Fallstudie zunächst nicht betrachtet. Aus diesem Grund wird nur die untere Verteilung (LPM) betrachtet. Die Streuungsbreite liegt bei 3 Verträgen und erreicht z.T. somit auch den Verlust von 11 Verträgen pro Jahr. Die Gleichung 3.1 beschreibt nun den Einfluss der technischen Infrastruktur mit den Schutzzielen (Confidentiality, Integrity, Availability) auf die Mächtigkeit der Menge der Verträge $|C|$.

$$\text{t-Var(C)} = \sigma_C^2 = \sum_{i=1}^{|C|} (c_i - E_w^T(C))^2 \cdot Pr\,(C = c_i). \tag{3.1}$$

Diese Erfahrungswerte der Fluktuation sind aus Beobachtungen der letzten 10 Jahre abgeleitet.

Für die Versicherung ist die Erreichbarkeit für ihre Kunden von sehr großer Bedeutung. Wenn die Kunden nicht per Telefon ihr Anliegen vorbringen können, sind diese ungehalten und bereit die Versicherung zu wechseln.

Bei dem Verlust der Vertraulichkeit entsteht ein Reputationsschaden für die Versicherung und es kündigen eine Reihe von Kunden. Es gibt interne Untersuchungen der Versicherungsfirma, die diesen Sachverhalt belegen. Dies ist einer der Gründe weshalb die Risikoanalyse durchgeführt wurde. Es hat sich gezeigt, dass durch die Risikoszenarien in der empirischen Fallstudie die Möglichkeit besteht, deutlich mehr Verträge zu verlieren, als durch den Fluktuationsschaden erwartet.

Die Abbildung 4 zeigt die LPM Verteilung in einer anderen Darstellung. Diese Darstellung wird sehr häufig für Verlustbetrachtungen vorgenommen. Es handelt sich um die diskreten Werte der Risikoszenarien, die in empirischen Fallstudien analysiert wurden. Es lässt sich zeigen, dass der erwartete Verlust zwischen 8 und 11 Verträgen deutlich überschritten wird. Es handelt sich also um einen unerwarteten Verlust.

Abschließend lässt sich für unsere Fallstudie festhalten, dass die Versicherung u.a. entschied, entsprechende Maßnahmen zur Verschlüsselung im Bereich des LAN und auf der Strecke in der MPLS-Cloud vorzunehmen. Dabei wurde darauf geachtet, dass die Kosten der Maßnahmen die möglichen Verluste nicht überschreiten.

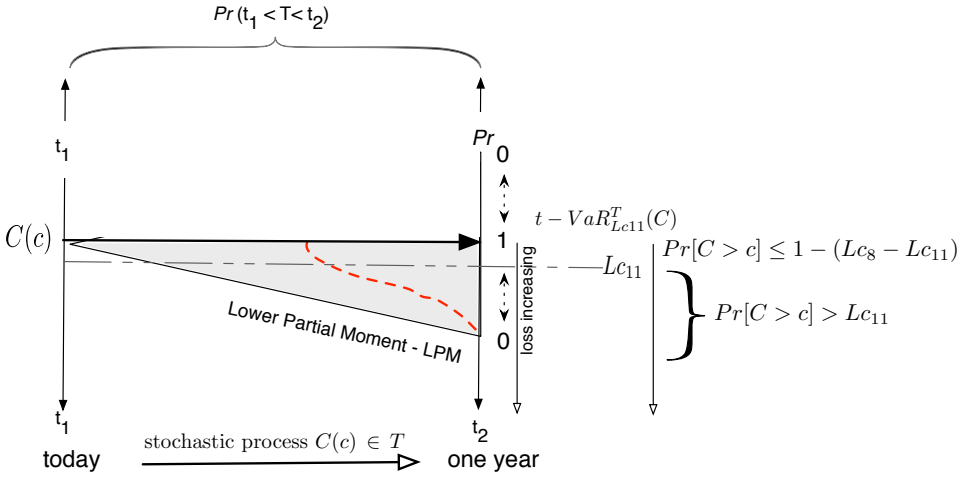


Abbildung 3: Technischer VaR für die Versicherung

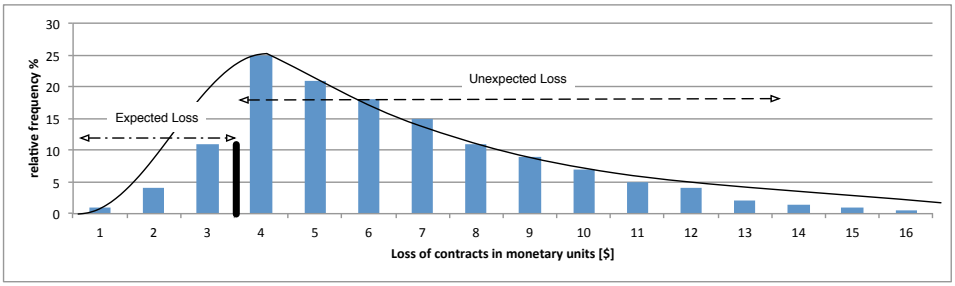


Abbildung 4: Verlustverteilung an Verträgen c_i durch den bekannt gewordenen Vertraulichkeitsverlust bei der VoIP-Telefonie

4 Kurzbeschreibung und weitere Untersuchungen

Es wird in diesem Artikel eine Methode vorgestellt, die einerseits einen technischen VaR ermittelt und andererseits auf die Besonderheiten einer IT-Infrastruktur eingeht. Am Beispiel der VoIP-Telefonie einer Versicherung wurde der t-VaR untersucht. Im Ergebnis lässt sich zeigen, dass der t-VaR eine adäquate Methode zur Bestimmung des VaR für den technischen Bereich im Rahmen der operationellen Risiken ist. Weiterführende Überlegungen zielen darauf ab für weitere Bereiche der IT-Infrastruktur einen t-VaR zu bestimmen, um dann eine Risikoaggregation zur Bestimmung des Gesamtrisikos im technischen Bereich durchzuführen.

Literatur

[BSI08] BSI. *BSI-Standard 100-2: IT-Grundschutz-Vorgehensweise*. Bundesamt für Sicherheit in der Informationstechnik, www.bsi.bund.de/gshb, 2.0. Auflage, 05 2008.

[EFK03] Paul Embrechts, Hansjörg Furrer und Roger Kaufmann. Quantifying Regulatory Capital for Operational Risk. *Derivatives Use, Trading and Regulation*, 9:217–233, 2003.

[Ing] T. R. Ingoldsby. *Fundamentals of Capabilities-based Attack Tree Analysis*. Amenaza Technologies Limited, 406 – 917 85th St SW, m/s 125.

[KMMS10] Barbara Kordy, Sjouke Mauw, Matthijs Melissen und Patrick Schweitzer. Attack-defense trees and two-player binary zero-sum extensive form games are equivalent. In *Proceedings of the First international conference on Decision and game theory for security*, GameSec'10, Seiten 245–256, Berlin, Heidelberg, 2010. Springer-Verlag.

[MEL01] A. P. Moore, R. J. Ellison und R. C. Linger. Attack Modeling for Information Security and Survivability. Technical Note CMU/SEI-2001-TN-001, Carnegie Mellon University, 2001.

[MHB85] Ali Mosleh, E. Richard Hilton und Peter S. Browne. Bayesian probabilistic rsika analysis. *ACM SIGMETRICS – Performance Evaluation Review*, 13, June 1985.

[MMRC+08] Ignacio J. Martinez-Moyano, Eliot Rich, Stephen Conrad, David F. Andersen und Thomas R. Stewart. A Behavioral Theory of Insider-Threat Risks: A System Dynamics Approach. *ACM Transactions on Modeling and Computer Simulation*, 18(02), April 2008.

[mOfSiIT05] Federal Office for Security in Information Technology. IT Baseline Protection Handbook. *Bundesanzeiger, Cologne*, 2003 - 2005.

[Sch99] B. Schneier. Attack Trees. *Dr. Dobb's Journal*, 24(12):21–29, 1999.

[SW04] O. Sheyner und J. Wing. *Tools for Generating and Anaylzing Attack Graphs*, Seiten 344–371. FMCO 2003, LNCS 3188. Springer-Verlag Berlin Heidelberg, 2004.

[Wei91] J. D. Weis. A System Security Engineering Process. *Proceedings of the 14th National. Computer Security Conference*, 1991.

5 Anhang

5.1 Grundlagen

Ausgehend von dem allgemeinen linearen Zusammenhang zwischen dem Risiko ($\mathcal{R}$), der Eintrittswahrscheinlichkeit (Pr) und dem monetären Ergebnis (I), wie in der Gleichung 5.1 ausgedrückt,

$$\mathcal{R} = Pr \times I \ \ [\mathbb{R}], \tag{5.1}$$

$$R_{sz} = E_p(b|S) \cdot I \tag{5.2}$$

$$R_{sz} = \left(\frac{E_p(b \cap S)}{E_p(b)} \right) \cdot I \tag{5.3}$$

$$\tag{5.4}$$

wird diese Form der Risikobetrachtung nicht verwendet, sondern in diesem Beitrag wird die Verlustvariante (Lower Partial Moment vgl. Abb. 4) herangezogen.

Wie diese in Gleichung 5.5 dargestellt ist, wird nachfolgend ganz allgemein für ein System Ψ gezeigt. Denn es wird für ein System nur das negative Ergebnis als Schaden [$\mathbb{R}^-$] betrachtet, wie es im Bereich des opRisk typisch ist (vgl. Embrecht et al. [EFK03]). Diese führt den möglichen Schaden (I) und den möglichen Eintritt eines negativen bedingten Ereignisses (Ep), d.h. einer existierenden Bedrohung (*thr*) unter der Bedingung (|), dass eine Schwachstelle (*vul*) existiert, zusammen. Die bedingte Wahrscheinlichkeit sagt etwas darüber aus, wie wahrscheinlich der Eintritt eines bereits bekannten Ereignisses ist, wenn ein (bestimmtes) anderes Ereignis bereits vorher eintrat. Mit dem Risiko als Maß geht dieses als Messgröße für einen Wahrscheinlichkeitsraum und der Verlustansatz (LDA) einher.

$$\mathcal{R} = \left(Pr_E \times L \right) [\mathbb{R}^-] \ \mapsto \Psi \tag{5.5}$$

In Gleichung 5.5 wird für die Eintrittswahrscheinlichkeit (Pr) eines Ereignisses (E) nicht die frequentistische Wahrscheinlichkeit, sondern die bedingte Wahrscheinlichkeit für ein Ereignis (Pr_E) gemäß dem Satz von Bayes verwendet.

Entsprechend der Bayes Statistik wird eine Hypothese aufgestellt, die hier besagt, dass eine Schwachstelle nur unter der Bedingung (|) von einer Bedrohung mit einem dazu passendem Bedrohungsprofil *(threat agent)* ausgenutzt werden kann. Oder vice versa, dass eine Bedrohung sich nur entfalten kann, wenn eine entsprechende Schwachstelle vorhanden ist. Die Bedrohung wird mit (*thr*) und die Schwachstelle mit (*vul*) bezeichnet. Somit kann die Gleichung 5.5 in die Gleichung 5.6 umgeformt werden.

$$\mathcal{R} = \left(Pr_E \left(vul \,|\, thr \right) \times L \right) [\mathbb{R}^-] \ \mapsto \Psi \tag{5.6}$$

Somit wird die bedingte Wahrscheinlichkeit, dass eine vorhandene Schwachstelle von ei-

ner Bedrohung ausgenutzt wird in der Gleichung 5.7 überführt mit

$$Pr_E(vul\,|\,thr) = \frac{Pr_E\,(thr \cap vul)}{Pr_E\,(thr)}. \tag{5.7}$$

Das heisst, der Zähler in der Gleichung 5.7 beschreibt die Schnittmenge zwischen der Menge der Bedrohungen und der Menge der Schwachstellen in der Wahrscheinlichkeitsebene Pr_E. Für das Zustandekommen der Schnittmenge wird gemäß der Bayes Statistik exogenes Wissen benötigt. Dieses notwendige exogene Wissen wird über Angriffsbäume und Angriffsprofile in die Wahrscheinlichkeitsebene gebracht, so dass sich damit die Wahrscheinlichkeit der Schnittmenge abschätzen lässt, wie wir in Abschnitt 5.2 zeigen.

Wird nun in die Definition 5.5 die Gleichung 5.7 eingesetzt, entsteht die nachfolgende Gleichung 5.8 für das System Ψ. Diese besagt, dass das Risiko ($\mathcal{R}$) einen Schaden ($L\ [\mathbb{R}^-]$) zu erleiden von dem Ereignis (E) abhängt, das wiederum von einer bedingten Wahrscheinlichkeit (Pr) für das Eintreten der Schnittmenge von Schwachstelle (vul) und Bedrohung (thr) abhängt.

$$\mathcal{R} = \left(\frac{Pr_E\,(thr \cap vul)}{Pr_E\,(thr)}\right) \times L\ [\mathbb{R}^-]\ \mapsto \Psi. \tag{5.8}$$

Aus der Gleichung 5.8 können einzelne Risikoszenarien $\mathcal{R} = \{R_{sz1}, ..., R_{szn}\}$ für verschiedene Systeme $\Psi = \{\psi_1, ..., \psi_n\}$ entwickelt werden, wie die nachfolgenden Gleichungen 5.9 bis 5.12 zeigen.

$$R_{sz1} = PrE_{p1}(vul_1\,|\,thr_1) \cdot l_{25c} \mapsto \text{VoIP-Telefonie} \tag{5.9}$$

$$R_{sz2} = PrE_{p2}(vul_1\,|\,thr_2) \cdot l_{25c} \mapsto \text{VoIP-Telefonie} \tag{5.10}$$

$$R_{sz2} = PrE_{p3}(vul_1\,|\,thr_3) \cdot l_{25c} \mapsto \text{VoIP-Telefonie} \tag{5.11}$$

$$\vdots$$

$$R_{szn} = PrE_{pn}(vul_n\,|\,thr_n) \cdot l_{25c} \mapsto \text{VoIP-Telefonie} \tag{5.12}$$

Die Wahrscheinlichkeit $PrE_{p1} - PrE_{pn}$ wird wie oben beschrieben mit der bedingten Wahrscheinlichkeit der Gl. 5.7 abgeschätzt.

Während Schwachstellen real vorhanden sind, sind Bedrohungen hypothetischer Natur. Um eine quantitative Aussage zur Risikolage einer Firma zu bekommen, ist es daher notwendig die Bedrohungen genauer zu analysieren. Denn nicht jede Bedrohung wirkt sofort und unmittelbar. Weiterhin wird vorausgesetzt, dass hinter jeder aktiven Bedrohung auch ein Akteur vorhanden sein muss. Historisch gesehen wurde zuerst 1991 von J.D. Weis das Konzept der Bedrohungsbäume (threat trees) von [Wei91] diskutiert.

5.2 Angriffsbäume und Bedrohungsprofile

Die Idee der Angriffsbäume (*attack trees*) geht auf die Arbeit von Weis [Wei91] zurück. In dem Artikel werden diese als logische Angriffsbäume beschrieben. Allgemein werden Angriffe mit graphischen Entscheidungsbäumen modelliert. Wenige Jahre später wird die

Idee der Bedrohungsbäume unter anderen von B. Schneier aufgegriffen und weiterentwickelt [Sch99]. Diese Arbeiten von B. Schneier führten zu weitreichenden Erweiterungen und Verbesserungen dieser Technik wie z.B. bei A.P. Moore et al. [MEL01] publiziert wurde. Etliche Tools sind entwickelt und publiziert worden, stellvertretend wird hier die Arbeit von [SW04, Ing] genannt; die Autoren geben einen Überblick über die Techniken und Tools. Die Gemeinsamkeit von Angriffsbäumen und Szenarienbetrachtung durch die Spieltheorie ist bei Kordy et al. in 2012 [KMMS10] heraus gearbeitet worden. Somit könnte ein ähnlicher Lösungsweg für die Bedrohungsszenarien und Bedrohungsprofilen mittels der Spieltheorie durchgeführt werden.

In dieser Arbeit wird die Idee von T. R. Ingoldsby [Ing] erweitert. Allgemein besitzen Bedrohungsbäume eine Wurzel bzw. Angriffsziel. Auf dieses Angriffsziel können verschiedene Zweige (Knoten) hinführen, die auch als Teilziele zu betrachten sind und jeweils von einem Blatt ausgehen. Hinter jedem Blatt verbirgt sich ein Angreifer mit unterschiedlichen Motiven. Die Blätter und Zweige werden gewichtet bei [Ing] und mit einem Akteur versehen. Die Gewichtung entspricht der Kriminellen Energie (Criminal Power, Cp) und erfolgt mittels drei Funktionen. Die Abschätzung der drei Funktionen spiegeln das Expertenwissen wider, das z.B. in der Bayesian Statistik erforderlich ist (vgl. Gleichung 5.7).

Bereits im Jahr 1985 haben A. Mosleh et al. sich Gedanken über die Bayesian Statistik in der Risikoanalyse gemacht [MHB85] und auf die beiden zentralen Verteilungen einer Risikoanaylse hingewiesen. Es ist zum einen die Verteilung des Eintretens eines Ereignisses. Da es sich um seltene Ereignisse handelt, wird hier die Poisson Verteilung vorgeschlagen, die später auch von vielen anderen Autoren verwendet wird. Die einparametrige Poisson Verteilung gibt gut die Abschätzung wieder, dass kleine Planabweichungen häufiger als größere Planabweichungen auftreten, wie z.B. in der Abb. 4 mit dem Lower Partial Moment (LPM) ausgedrückt wird. Die Herausforderung besteht nun darin eine gute Abschätzung für die Parameter der Verteilung zu bekommen. Diese Abschätzung wird in dieser Arbeit über die Bedrohungen und Angriffsbäume in Kombination mit der Funktion der Kriminellen Energie vorgenommen.

Auf der anderen Seite wird in dem Artikel von A. Mosleh et al. [MHB85] die Verlustverteilung durch die Log Normal Verteilung approximiert. Auch diese Verteilung entsprach für lange Zeit der Vorstellung, dass kleine Verluste häufiger als große Verluste auftreten. Diese Vorstellung wurde durch die Black Swan Theorie (Extrem Wert Theorie) revidiert und wird nun häufig durch eine Generalisierte Pareto Verteilung (GPD) ersetzt. Oftmals ist die Verlustverteilung einfacher zu bestimmen als die Häufigkeitsverteilung, wenn die Auswirkungen (Schaden) auf die Geschäftsprozesse bezogen werden.

Die Risikofunktion der kriminellen Energie (Cp) entspricht der *criminal function*, die wiederum additiv durch drei Funktionen zusammengesetzt wird, repräsentiert das Expertenwissen für die Bayesian Risikoanalyse (vgl. Gleichung 5.9 - 5.12). Die Kriminelle Energie wird durch die Kostenfunktion *(cost of attack)*, die technische Machbarkeitsfunktion *(technical function)* und die Bemerkbarkeitsfunktion *(noticeability function)* repräsentiert. Die drei Funktionen sind bereits bei T. R. Ingoldsby [Ing] erwähnt und müssen der jeweiligen Untersuchung (Inspektion) angepasst werden.

Die Abbildung 5 besagt, dass ein Akteur für einen Angriff auf die VoIP-Telefonie der Ver-

sicherung bereit ist (Willingness to spend) Geld für tools auszugeben. Diese Bereitschaft bewegt sich zwischen 0 - 1 (Ordinate) sinkt mit zunehmenden Kosten (Abzisse). Dies lässt sich einfach erklären, da im Internet eine ganze Reihe von kostenfreien Werkzeugen (tools) zu finden sind, die alle gut geeignet sind eine VoIP-Telefonie zu bedrohen.

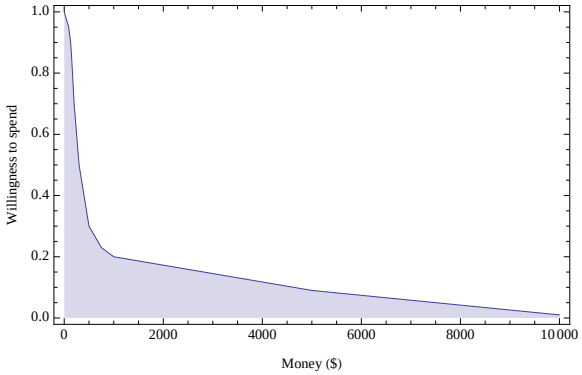


Abbildung 5: Funktion der Kosten für einen Angriff

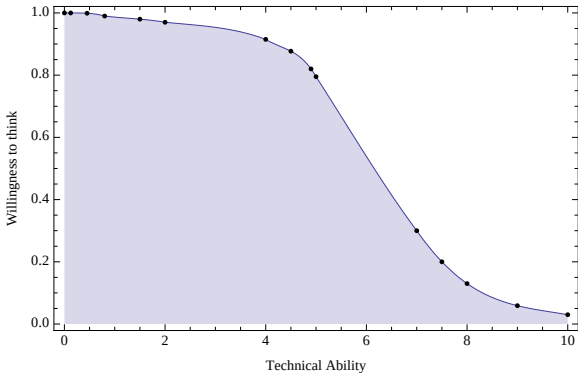


Abbildung 6: Funktion der zu verwendenden Technologie

Die Abb. 7 zeigt die Bemerkbarkeitsfunktion, die ausdrückt wie ein Akteur seinen Angriff verschleiern will, damit er nicht entdeckt wird. Aus diesen drei Funktionen ist die Bedrohung eines Angriffs näher zu bestimmen. Diese wird als kriminelle Energie bezeichnet. Diese Funktionen müssen jeweils auf die Situation angepasst werden und spiegeln das exogene Wissen wider, das bei der bedingten Wahrscheinlichkeit notwendig ist.

Als ein Beispiel für die technischen Möglichkeiten sei ein Wert von 0,5 auf einer Skala von 1.0 Werten und ein Wert von 0,3 auf der Bemerkbarkeitsfunktion verzeichnet. Wenn

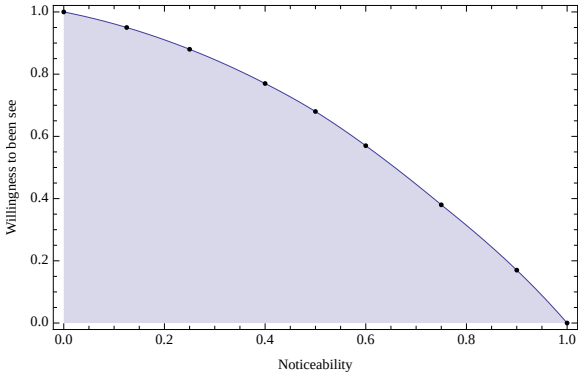


Abbildung 7: Funktion der Entdeckung

diese Werte verwendet werden, so ergeben sich die folgenden Werte:

$$f_{cost}\,(25) = 0.9 \tag{5.13}$$

$$f_{tech\,ability}\,(05) = 0.9 \tag{5.14}$$

$$f_{noticeabiity}\,(0.3) = 0.85 \tag{5.15}$$

und daraus die kriminelle Energie, die als Risikofunktion verstanden wird.

$$CE = f_{cost} \cdot f_{tech\,ability} \cdot f_{noticeabiity} \tag{5.16}$$

$$CE = 0.6885 = 0.9 \cdot 0.9 \cdot 0.85 \tag{5.17}$$

Die Diskussion über die Motivation und Vorteile eines Angreifers wird in diesem Artikel nicht behandelt.

Envisioning Smart Building Botnets

Steffen Wendzel[1], Viviane Zwanger[1,2], Michael Meier[1,2], Sebastian Szlósarczyk[1,2]

[1]Cyber Defense Research Group, Fraunhofer FKIE
[2]Institute of Computer Science 4, University of Bonn
{wendzel,zwanger,meier,szlos}@cs.uni-bonn.de

Abstract: A building automation system (BAS) is the IT equipment within a building that monitors and controls the building (e.g., measuring temperature in a room to configure the heating level within the same room). We discuss the potential and the use of botnets in the context of BAS. Our botnet concept and scenario is novel in the sense that it takes advantage of the phyiscal capabilities of a building and as it has to adapt to a specialized environment being highly deterministic, predictable, simplistic and conservative. These properties make anomalies easy to detect. *Smart building botnets* allow the monitoring and remote control of (critical) building automation infrastructure in public and private facilities, such as airports or hospitals. We discuss why building automation botnets could thus enable attackers to cause various critical damage on whole regions and economies. Hiding the command and control communication is a highly beneficial step to adapt botnets to the BAS environment. We show that this is not necessarily a big hurdle and can be solved using existing covert channel techniques.

1 Introduction

This paper combines three research areas, namely building automation research, botnet research, and network covert channels.

Building automation systems (BAS) are IT components integrated in and capable of controling and monitoring buildings. BAS aim at improving the energy efficiency of houses, at increasing the comfort and safety for people living or working in a building, and at decreasing a building's operation costs. Therefore, it is necessary to enable a BAS to operate critical equipment like smoke detectors or physical access control components.

Botnets have become an essential and indispensable part of todays' criminal infrastructure. Botnets allow for controlling and drawing profit not only from individual computer systems but also through mass infection. The modern criminal botnets have become extremely complex and grown into a robust ecosystem of organized crime with a sophisticated service landscape [GBC+12, CGKP11]. The spectrum of criminal activity ranges from hosting stolen credit card information, selling private data in large chunks as well as utilizing cheap computing power (also used for legitimate purposes), to even directly blackmail victims (i.e. encrypt hundreds of computer systems and extort a ransom from victims).

To criminals, BAS offer a completely new market, with new opportunities of drawing profit on a big scale. Thus, BAS are likely to be discovered for the criminal market and adapted for use within the current botnet infrastructure. However, the adaptation of BAS for the use in botnets requires clearing a few hurdles. Our intention is to show that these hurdles are minimal, offering only an initial barrier and short time window before being tackled by criminals. One of these barriers is, as we will show in the following, the need to hide the information flow of a botnet due to the deterministic and specialized nature of BAS. Among other things this requires the use of modern covert channel techniques.

Covert channels were defined by Lampson as channels "*not intended for information transfer at all*" in 1973. The definition of covert channels was extended by the U.S. Department of Defense (DoD) as "*any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy*" [Dep85]. In other words, covert channels are not foreseen by a system's design but due to their existence nevertheless allow a security policy-breaking communication [And08], e.g. between confined processes. Covert channels in networks enable stealthy policy-breaking communication between computers and help adversaries to keep a low profile.

We present the first work combining the three mentioned research areas to highlight the potential of a novel class of botnets, namely *smart building botnets*, i.e. botnets that do not attack common network systems to utilize their computing power and network connection, but BAS instead, aiming to utilize their sensors and actuators to perform physical measurements and actions. Thus, smart building botnets allow the monitoring and control of buildings. According to the botnet's size, entire smart cities or even economies could theoretically become part of a smart building botnet. Moreover, BAS communication does not differ between most buildings as only few BAS protocols are widely used. Critical building infrastructure such as hospitals or airports can thus become part of a smart building botnet as well as private housing facilities.

The benefits for malware developers are manifold. First, malware attackers could monitor events (e.g. movement patterns) in a large number of buildings and could thus create usage profiles of inhabitants, which could be sold later on a black market. Second, miscreants can aim at causing a denial-of-service in a building (e.g. forcing an evacuation by a false fire alarm). Third, in contrast to mobile devices and PC systems, BAS are permanently available, rarely modified, face nearly no security features, are designed for long-term deployment and are rarely patched. This makes them an excellent choice for placing bots. Fourth, buildings can be used to blackmail their inhabitants and owners (e.g. forcing the transfer of money to a bank account to end a disruption on a critical system such as an airport baggage transfer system or lifts in a hospital).

Roadmap. The paper is structured as follows. Section 2 covers related work on covert channels, botnet communication, and building automation. Section 3 explains the concept of smart building botnets, their benefits for malware authors, and their technical feasibility. We discuss preventive measures to be taken in order to protect building infrastructure and summarize our findings in Section 4.

2 Related Work

We combine three research areas, namely network covert channels, botnets, and BAS. Related work on all three areas is discussed separately.

Recent Developments in Covert Channel Research: Yarochkin *et al.* propose *adaptive covert channels* [YDL+08], i.e. covert channels able to adapt their communication techniques to changes in the network environment. For instance, if a covert channel communicates using HTTP and an administrator blocks all HTTP communication after some time, the channel can switch to another protocol, e.g. DNS, to continue its operation.
Other significant developments of the last years are the so-called *control protocols* or *micro protocols* [RM08]. Control protocols consist of small headers placed inside the hidden data transferred through the covert channel. Therefore, most control protocols are placed in unused fields of network headers. Alternative approaches exist, e.g. [MK06] place part of the hidden control data in a watermark.
Control protocols enable various features such as reliable data transfer, session management, peer discovery, dynamic overlay routing, switching of the utilized network protocol, and adaptiveness to changes in the network configuration [WK14].

Modern Botnet Communications: Former botnets transferred their command and control (C&C) traffic in a simple manner using network protocols such as IRC. Today, botnets obfuscate and encrypt C&C traffic in order to compound detection and take over. Recently, a Linux malware using a simple network covert channel within SSH connections has been discovered by Symantec [Sym13]. Moreover, a steganographic botnet named *StegoBot* already exists. However, StegoBot is not based on network covert channels and rather hides within social network communication by using image steganography [NHP+11]. Another example for this type of covert channels is the *Feederbot*, described by Dietrich *et al.* [DRF+11] which uses DNS as hidden C&C communication channel. Using social networks for covert channel C&C communication is a rising trend in the criminal landscape, as shown by Kartaltepe *et al.* [KMXS10] who described an example "social" botnet using *Twitter*. These examples underline that the integration of information hiding features into malware has already started.
We see a huge potential for attackers willing to improve the stealthiness of C&C communication using covert channels. Micro protocols can therefore replace existing C&C protocols, as existing micro protocol engineering approaches (cf. [WK14]) do not only optimize the protocol's stealthiness but in addition help to keep the botnet communication feature-rich. Adaptive covert channels can additionally increase the robustness of C&C communications as they can bypass blocked protocols and, in combination with dynamic overlay routing, bypass firewall routers.

Building Automation (In)Security: BAS form networks which can be interconnected with other buildings and the Internet (e.g., for remote monitoring purposes) and therefore use different protocols, especially *Building Automation Control and Network* (BACnet), *European Installation Bus* (EIB)/*Konnex* (KNX), and *Local Operating Network* (LON). These protocols feature specific security standards, which have been improved over the time. However, even though security features are available in standards they are commonly not integrated in devices nor used in practice.

Former BAS were designed to work as isolated stand-alone systems with basically no security features. Due to the need to increase BAS' functionality, inter-connectivity, interoperability, and especially Internet access became significant features of BAS. Unfortunately, precisely the inter-connectivity of BAS enables remote attacks. Attacks on BAS can, for instance, focus on getting physical access to a building by exploiting window or door actuators [Hol03], on getting access to an organizational intranet [SZ12], or on disabling a building's functionality via DoS attacks [GKNP06].

Granzer *et al.* presented a hierarchical attack model for BAS in which they distinguish attacks on the BAS network from attacks on the devices themselves. Network attacks comprise the interception of a communication (using a sniffer), the modification of network data (via man-in-the-middle attacks), the interruption of the communication (e.g., denial of service attacks and the redirection of traffic) and fabrication attacks (i.e., the generation of new malicious frames and replay attacks) [GPK10]. As for device attacks, the authors distinguish between software-side attacks (e.g., code injection), side channel attacks (based on timing, power, and fault behavior analysis), and physical attacks (e.g., replacement of devices) [GPK10].

While Granzer *et al.* were the first authors to *mention* side channel attacks, our own previous work describes the existence and use of covert and side channels *within* BAS (cf. [Wen12, WKR12]). Side channels within BAS allow inhouse adversaries to monitor events in the building. For instance, an employee could try to monitor events in floors or areas of a building he has no access to. Covert channels in BAS allow inhouse adversaries to bypass data leakage protection (DLP) and to leak confidential data through the BAS out to the Internet. However, none of these mentioned publications covers the concept of smart building botnets.

3 Covert Smart Building Botnets

Existing botnets take over IT systems in order to utilize the performance, storage space, and network connection of these systems to attack target systems or networks (DoS attacks) or to transfer spam messages. Physical devices, like temperature sensors of computers, are not used by today's botnets. If bots attack BAS, two different scenarios may arise. First, the remote accessible component of the BAS (e.g. an embedded Linux system) is attacked and used for already known purposes (e.g. spam transfer). Secondly, the actual capabilities of the BAS are used by the bot software. We focus on the latter case as it represents a novel approach.

3.1 Utilization of BAS Equipment for Bots

BAS comprise sensors (e.g. temperature, humidity, or presence sensors) and actuators (e.g. electronic windows, heating, ventilation, air-conditioning, or electronic light switches). These sensors and actuators allow two generic uses for adversaries:

1. *Monitoring of Events in Buildings*: Spyware can determine whether persons are present at a particular location of the building (e.g., based on temperature and heating in rooms as well as by using presence sensors). Intruders can use such information to organize and direct break-ins. Therefore, thefts could, for instance, steal equipment on floor 2 if all persons reside or move around on other floors or in the basement. Moreover, monitoring data such as movement patterns of inhabitants can be leaked to the botmaster using a network covert channel.

2. *Remote Control of Buildings*: Malware can take advantage of the actuators of a building, e.g. heating, air-conditioning, ventilation or elevators. Even for single BAS, miscreants may cause considerable damage using these actuators [Fis12] (e.g., disabling fire alarms before placing a fire in the building [Hol03], activating a fire alarm at an airport to cause chaos [Con08], or deactivating an airport's luggage transport system). Another example would be to cause a DoS at the physical access control systems of an enterprise building to prevent that employees can access their office rooms, what could, in the worst case, result in inactivity of a company.

However, speaking of a *botnet*, new scenarios arise, namely the attack of larger infrastructure distributed over many buildings, smart cities, or, theoretically, even economies or states. If adversaries manage to place a high number of building automation bots in such a smart environment, the possible effects of the discussed scenarios increase, as we illustrate using the following scenarios.

1. The mentioned actuator-based attacks (e.g., attacking physical access control systems) can be multiplied by a botnet, e.g. miscreants may attack multiple airports, train stations, and public or private buildings of an economy simultaneously, i.e. the scale of the attack is extended.

2. In order to increase sales for oil or gas, an oil (gas) company could place bots in the BAS of a region or economy and could slightly but permanently increase the heating, ventilation, and air-conditioning levels at night in as many buildings as possible. This would increase the energy consumption and the need for oil (gas).

3. Hellwig *et al.* show that excessive heating affects people's efficiency and capability to work within buildings [HNB+12]. Smart building botnets could take advantage of this effect by increasing the temperature in trading places (e.g. at the Frankfurt Stock Exchange). This allows for influencing trader's reaction time on the targeted stock exchanges.

4. In [Con08], it has been reported that turning off the heating of a server room can cause a room-wide denial of service due to server failures. Such a scenario can be extended by BAS bots to attack a high number of server rooms (simultaneously).

5. An assailant could try to blackmail inhabitants of a high number of buildings by attacking these buildings. Elders, handicapped and weak people could be locked inside buildings (by closing windows and doors automatically) if they do not transfer an amount of money to a given bank account. While, at first sight, elders appear

as untypical BAS users, the value of BAS integration in their homes is linked to *ambient assisted living* (AAL), i.e. the support of the daily life of inhabitants. AAL allows elders to live longer in their own homes before being forced to move to a nursing home and thus AAL is a growing market.

6. Due to the high number of sensors integrated in modern buildings and their diversity, assailants can aim at collecting as many personal data of inhabitants, employees etc. within a building as possible. For instance, presence sensors, temperature sensors, heating levels and light actuator states indicate the (potential) presence of persons in a given area of a building while the nightly use of light in the bathroom may indicate the illness of an inhabitant. Moreover, the presence of AAL sensors and actuators can indicate serious health problems. Such information can be sold at the black market and, for instance, allow thiefs to prepare intrusions in a way adapted to the movement patterns in a building or to insurance companies for allowing an a priori healthiness check of future insurants.

7. A coordinated smart building botnet could activate a high number of devices (especially white goods and other household appliances) simultaneously in order to create peaks in a region's energy consumption.

However, a direct interaction with the BAS can raise a higher level of attention as direct control information passes public networks and because BAS control information is not hidden. If, on the other hand, BAS control information is hidden in low-attention raising covert channels, the operations can be kept more stealthy.

3.2 Botnet Architecture and Feasibility in Practice

The concept of a smart building botnet is visualized in Fig. 1. The botmaster controls a number of BAS using network covert channels and applies the previously introduced techniques (control protocols, adaptive covert channels) to hide his communication. The sensors and actuators of each BAS are under control and monitoring of the particular bot.

Bots are therefore placed directly on the Internet gateways of BAS or on remotely accessible *central control units* (CCUs) used in cheaper home equipment like the *HomeMatic*. An increasing number of these cheap home automation systems become available to end-users and thus increase the number of so-called *smart homes*. With the rising number of smart homes, the number of attackable private houses as well as the effect of bot attacks on BAS increases. Typical CCUs run on embedded systems (e.g. embedded Linux, as in case of HomeMatic) and are, even if set up in a secure way, not patched regularly by private owners.

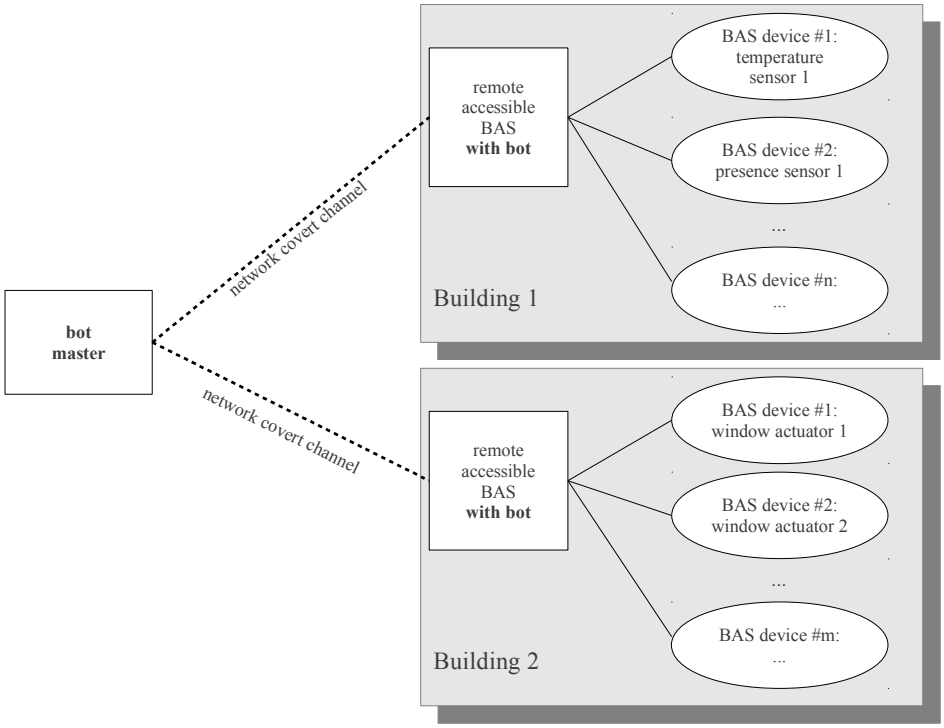


Figure 1: Concept of building automation botnets in combination with covert C&C channels.

A similar problem arises in public and business buildings. Although hardware components are usually of higher quality and robustness, the patching of BAS equipment often is not done by the administration staff. Reasons, why patches are not applied, are the lack of available patches, the limited capabilities of legacy equipment, and the lack of the staff's security awareness.

The infection of BAS installations with bots can be achieved in three ways. First, BAS can be infected by searching vulnerable BAS via the computer search engine *SHODAN* (cf. *www.shodanhq.com*). As shown in the *Industrial Risk Assessment Map* (IRAM), control systems and BAS with Internet connectivity can be found worldwide. In the USA, BAS are available even more often than SCADA systems, PLC systems, HMIs and other automation equipment [MK14]. The IRAM project moreover determined the presence of approximately 14.500 directly online accessible BAS in the USA and approximately 1.300 BAS in Germany. Of the determined remotely accessible BAS installations, 9% are linked to officially known CVE vulnerabilities [MK14].

Secondly, the infection of BAS can be done by hand. Therefore, BAS wardriving, which we presented in [KW13], is considered a means to infiltrate a smart city. In BAS wardriving, adversaries drive through a city in order to determine the presence of wireless BAS. Wireless BAS are popular as their integration in existing buildings is easy and as they are

cheaper than wired BAS. However, BAS wardriving is *slow* since only semi-automated (attackers need to invest time to drive through a city). Moreover, no numbers on the quantity of vulnerable wireless BAS per region are known. Therefore, we consider the second approach as unsuitable for the setup of botnets. In future work, we plan to investigate the feasibility of utilizing GPS-enabled smart phones to discover wireless BAS within a region. Combining the utilization of smart phones with a vulnerability test of BAS would change the second approach from a semi-automated to an automated approach.

A third approach exist in which no bot software must be placed at the target system and in which no network covert channel needs to be applied. Like in the first approach, an attacker searches for directly accessible BAS installations using SHODAN, but then probes whether these BAS directly accept BAS protocol commands. For instance, a BACnet device could be directly connected to the Internet and could thus accept BACnet commands. Such an accessible BACnet device can execute monitoring commands as well as actuator commands and thus, needs no bot software and can be directly controlled by the botmaster. The drawback of the third approach is that it is more likely to raise attention than the covert channel-based first approach.

We consider mixing all three approaches as a useful strategy for malware authors. A smart building botnet might comprise directly controlled BAS (third approach) as well as covertly controlled BAS (first approach). The integration of manually selected BAS (e.g. using BAS wardriving) into a botnet appears only beneficial if the BAS is of significant importance (e.g. an airport may increases the magnitude of a miscreant's attack).

4 Discussion and Conclusion

The delineated concept of smart building botnets and their related risks lead to the question of how to react to this upcoming threat. After answering this question, we summarize our work.

4.1 Recommendations for next steps to be taken

We believe that the research community should continue to investigate botnets for smart environments and propose further countermeasures to BAS vendors, BAS owners, BAS operators and to the public.

BAS vendors, on the other hand, need to urgently supply their existing BAS equipment with better protection (hardware as well as software, especially for gateway systems). A challenge in this regard is the migration and hardening of legacy BAS. The migration process includes the integration of newer BAS communication protocols linked to better security features.

It is probably not feasible to raise public awareness for BAS threats in a short term, especially not for the concept of smart building botnets. Furthermore, we believe that the

security awareness and responsibilities of BAS operators and BAS owners will be hard to raise. In other words, we must appeal to the research community, to the politics and to the BAS industry for achieving a better protection on behalf of BAS users.

Therefore, academic research, politics and industrial development need to focus two aspects: *i)* creating means for the protection of existing and upcoming BAS; *ii)* creating means for the detection and elimination of bot infections in BAS.

Moreover, the results are required to be delivered in a limited time window: we believe that the organized crime will use any available means to make profit. It is thus only a matter of time before the hurdles of mastering BAS environments are taken to implement smart building botnets. If no further defensive measures are delivered, botnets will gain stable ground in the BAS infrastructure.

4.2 Summary

In this paper, we provide an outlook for the extension of current botnet techniques to building automation systems (BAS) by presenting the concept of *smart building botnets*. Smart building botnets utilize the physical capabilities (sensors and actuators) to monitor and remotely control BAS. We reason that such botnets are feasible in practice and linked to various benefits for malware authors.

Our reason for depicting the threat of smart building botnets is to speed up the integration of better protection means by vendors of BAS equipment. We believe that the time window available till such botnets become common practice must be used to enhance the security of existing and upcoming BAS.

Future work will focus on the evaluation of our concept using different BAS technologies as well as on the development of means to protect BAS.

References

[And08] R. Anderson. *Security Engineering - A Guide to Building Dependable Distributed Systems*. Wiley, 2 edition, 2008.

[CGKP11] J. Caballero, C. Grier, C. Kreibich, and V. Paxson. Measuring Pay-per-Install: The Commoditization of Malware Distribution. In *USENIX Security Symposium*, 2011.

[Con08] R. Condon/ComputerWeekly. New-generation building management systems blow a hole in security, 2008. http://www.computerweekly.com/news/1331080/New-generation-building-management-systems-blow-a-hole-in-security, retrieved: December 2013.

[Dep85] Department of Defense. Trusted Computer System Evaluation Criteria (TCSEC, DoD 5200.28-STD, Orange Book), 1985.

[DRF+11] C. J. Dietrich, C. Rossow, F. C. Freiling, H. Bos, M. v. Steen, and N. Pohlmann. On Botnets that use DNS for Command and Control. In *Proceedings of EC2ND'11*, Gothenburg, Sweden, September 2011.

[Fis12] D. Fisk. Cyber security, building automation, and the intelligent building. *Intelligent Buildings International*, 4(3):169–181, 2012.

[GBC+12] C. Grier, L. Ballard, J. Caballero, N. Chachra, C. J. Dietrich, et al. Manufacturing compromise: the emergence of exploit-as-a-service. In *ACM Conference on Computer and Communications Security*, pages 821–832, 2012.

[GKNP06] W. Granzer, W. Kastner, G. Neugschwandtner, and F. Praus. Security in networked building automation systems. In *Proc. 2006 IEEE International Workshop on Factory Communication Systems*, pages 283–292, 2006.

[GPK10] W. Granzer, F. Praus, and W. Kastner. Security in Building Automation Systems. *IEEE Transactions on Industrial Electronics*, 57(11):3622–3630, November 2010.

[HNB+12] R. T. Hellwig, I. Nöske, S. Brasche, Hj. Gebhardt, I. Levchuk, and W. Bischof. Hitzebeanspruchung und Leistungsfähigkeit in Büroräumen bei erhöhten Außentemperaturen. Abschlussbericht zum Projekt HESO, Bundesanstalt für Arbeitsschutz und Arbeitsmedizin, 2012. (in German).

[Hol03] D. G. Holmberg. Enemies at the Gates. *BACnet Today (A Supplement to ASHRAE Journal)*, pages B24–B28, 2003.

[KMXS10] E. J. Kartaltepe, J. A. Morales, S. Xu, and R. S. Sandhu. Social Network-Based Botnet Command-and-Control: Emerging Threats and Countermeasures. In *8th International Conference on Applied Cryptography and Network Security (ACNS)*, volume 6123 of *LNCS*, pages 511–528, 2010.

[KW13] B. Kahler and S. Wendzel. How to own a Building? Wardriving gegen die Gebäudeautomation. In *Beitrge zum 20. DFN Workshop zur Sicherheit in vernetzten Systemen*, pages H1–H13, 2013. (in German).

[MK06] W. Mazurczyk and Z. Kotulski. New security and control protocol for VoIP based on steganography and digital watermarking. *Annales UMCS, Informatica*, AI 5:417–426, 2006.

[MK14] J.-O. Malchow and J. Klick. Erreichbarkeit von digitalen Steuergeräten – ein Lagebild. In *Beitrge zum 21. DFN Workshop zur Sicherheit in vernetzten Systemen*, 2014. (in German, to appear).

[NHP+11] S. Nagaraja, A. Houmansadr, P. Piyawongwisal, V. Singh, P. Agarwal, and N. Borisov. Stegobot: A Covert Social Network Botnet. In *Proceedings of the 13th International Conference on Information Hiding (IH'11)*, pages 299–313, Berlin, Heidelberg, 2011. Springer-Verlag.

[RM08] B. Ray and S. Mishra. A Protocol for Building Secure and Reliable Covert Channel. In *Proc. 6th Annual Conference on Privacy, Security and Trust (PST 2008)*, pages 246–253. IEEE, 2008.

[Sym13] Symantec. Linux Back Door Uses Covert Communication Protocol, 2013. http://www.symantec.com/connect/blogs/linux-back-door-uses-covert-communication-protocol, retrieved: December 2013.

[SZ12] S. Soucek and G. Zucker. Current developments and challenges in building automation. *e & i (Elektrotechnik und Informationstechnik)*, 129(4):278–285, 2012.

[Wen12] S. Wendzel. Covert and Side Channels in Buildings and the Prototype of a Building-aware Active Warden. In *IEEE International Workshop on Security and Forensics in Communication Systems (SFCS 2012)*, pages 6753–6758, 2012.

[WK14] S. Wendzel and J. Keller. Hidden and Under Control: A Survey and Outlook on Covert Channel-internal Control Protocols. *Annals of Telecommunications (ANTE)*, 69(3-4), 2014. (to appear).

[WKR12] S. Wendzel, B. Kahler, and T. Rist. Covert Channels and their Prevention in Building Automation Protocols – A Prototype Exemplified Using BACnet. In *Proc. 2nd Workshop on Security of Systems and Software Resiliency*, pages 731–736. IEEE, 2012.

[YDL+08] F. V. Yarochkin, S.-Y. Dai, C.-H. Lin, et al. Towards Adaptive Covert Communication System. In *Proc. 14th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2008)*, pages 153–159. IEEE Computer Society, 2008.

Forensic Zero-Knowledge Event Reconstruction on Filesystem Metadata

Sven Kälber, Andreas Dewald, Steffen Idler
Department of Computer Science
Friedrich-Alexander-University Erlangen (FAU)
Martensstr. 3
91058 Erlangen, Germany

Abstract: Criminal investigations today can hardly be imagined without the forensic analysis of digital devices, regardless of whether it is a desktop computer, a mobile phone, or a navigation system. This not only holds true for cases of cybercrime, but also for traditional delicts such as murder or blackmail, and also private corporate investigations rely on digital forensics. This leads to an increasing number of cases with an ever-growing amount of data, that exceeds the capacity of the forensic experts. To support investigators to work more efficiently, we introduce a novel approach to automatically reconstruct events that previously occurred on the examined system and to provide a quick overview to the investigator as a starting point for further investigation. In contrast to the few existing approaches, our solution does not rely on any previously profiled system behavior or knowledge about specific applications, log files, or file formats. We further present a prototype implementation of our so-called zero knowledge event reconstruction approach, that solely tries to make sense of characteristic structures in file system metadata such as file- and folder-names and timestamps.

1 Introduction

The data deluge, investigators are facing in digital forensic investigations nowadays, is remarkable and often leads to a time-consuming analysis. Selective imaging [Tur06, SDF13] is considered a possibility of reducing the amount of data to be processed, while abstraction, as well as automation, are regarded to be necessary for investigation speedup [Gar10]. Well known investigative process models [Cas11, Car05] yield starting points for automation during acquisition and extraction of digital evidence and during event reconstruction. Today, commonly used tools already provide automation for the acquisition and extraction of digital evidence, but lack features for automatic event reconstruction.

1.1 Motivation

Up to now, there are only few approaches to automated event reconstruction. One approach is to identify characteristic patterns (digital fingerprints) that emerge in filesystem timestamp metadata while running a specific application or action [KDF13, JGZ11]. Another

approach is having individual parsers or analyzers that operate on specific application logfiles and data stores such as sqlite databases [HP12]. While the approach of fingerprinting is more generic, since it is based only on filesystem metadata, the approach of logfile parsing is more specific because each application implements its own logfile format. Thus, investigators are in need of an individual parser for every application of interest. More importantly, the logfile format might change over time, therefore multiple parsers may be needed for different appliction versions. The concept of digital fingerprinting on the other side, similar to classical forensic fingerprint comparison, requires a previously aquired set of known fingerprints. For this reason, prior to actually matching fingerprints in a digital investiagtion, a costly and time consuming profiling phase for each application is required.

1.2 Contributions and Outline

In this paper, we examine the feasibility of automatic event reconstruction that spares prior knowledge of application specifics. We call this approach *zero-knowledge event reconstruction*, since it does not depend on any previous profiling or logfile parsers.

Our approach is to initially extract all filesystem timestamps of a given storage device, then cluster accumulations of timestamp values around different points in time to events. Based on storage location, file names and types, as well as the type of timestamp, these events are then labeled on a best guess basis. To support the investigator during the investigation and to provide a quick overview of all events found this way, those events are being visualized in a dynamically browsable graphic timeline. Moreover, the timeline contains details about which timestamp pattern led to the identification of each specific event.

In particular, we make the following contributions:

- We introduce a novel approach to automated event reconstruction that does not require training phases or application specific analyzers. (Section 3)
- We give a prototype implementation of our approach called *0KER* that identifies events based on timestamp information stored in the filesystem. (Section 4)
- We evaluate the chances and limitations of our approach using the prototype implementation and a compare it to our former approach. (Section 5)

1.3 Related Work

James et al. [JGZ11] generate timestamp signatures for identifying application startups in post-mortem digital investigations. Those signatures are created by first monitoring file access and the Windows Registry with the Microsoft Process Monitor during startup. In a second step, the timestamp update behavior for all files identified with the Process Monitor, is then monitored and saved a signature for the corresponding application startup.

In previous work [KDF13], we have developed a forensic fingerprinting framework named

Py3xF that allows investigators to generate fingerprints for automated event reconstruction. We focused not only on the startup process of different applications but also on application specific actions like sending an email / instant message or bookmarking a website. *Py3xF* allows to automatically create application profiles based on timestamp modifications that occur during application runtime. Based on these application profiles, fingerprints are generated that consist of only unique timestamp modifications. Those fingerprints can than be matched against extracted metadata of a filesystem. All matching patterns are then reported back to the investigator. This way, only known and already profiled / fingerprinted applications and actions may be identified and reported.

Gudjonsson [Gud13] developed the well-known super-timelining tool *log2timeline*. It parses known log files such as windows event logs or apache web server logs. Results can be visualized in timelines using different timeline interfaces like Computer Forenics TimeLab, BeeDocs or SIMILE. However, this approach is very application specific, since individual log file parsers need to be implemented. Moreover, investigators might still be confronted with an overwhelming amount of information, since *log2timeline* does not automatically cluster multiple low-level events to a smaller amount of more meaningful high-level events.

Hargreaves and Patterson [HP12] present an approach for reducing the amount of data by identification of high-level events in automated timeline reconstruction. *PyDFT*, their implementation of a digital forensic timeline generator operates in two stages. In the low-level event extraction stage, filesystem timestamps and timestamps from log files and registry hives are extracted. In the high-level event reconstruction stage, specific analyzers are used to re-create more meaningful high-level events such as Google searches and USB device connection. This approach reduces the data deluge that investigators are facing with tools like *log2timeline* but additionally to extractors, handcrafted sophisticated analyzers are required.

After we have now discussed related work in the field of automated event reconstruction and noticed that all approaches depend on prior knowledge of application specifics in either handmade parsers / analyzers or automatically profiled / fingerprinted timestamp patterns, we detail our approach of *zero-knowledge event reconstruction* including some required background information in the following sections.

2 Background

The basic idea of this work is to extract the timestamp information stored in the filesystem of a seized hard disk and then try to automatically reconstruct certain events (such as application startups) that recently happened on the system it was taken from. This is done by clustering all files that were accessed, modified or created around certain points in time. Since we set our focus on the NTFS filesystem, we will now briefly describe the specifics of NTFS timestamps before we introduce the applied string metric and clustering algorithms that are used for event reconstruction.

2.1 Timestamps in the NTFS filesystem

Microsoft's New Technology File System (NTFS) is used as the default filesystem in Windows NT, 2000, XP, Vista, 7 and 8, making it the most widely used filesystem on private computers nowadays. Besides other metadata, like the filename and size, NTFS keeps track of the four following timestamps:

- atime: the time, a file / folder was last accessed.
- mtime: the time, a file / folder was last modified.
- crtime: the time, a file / folder has been created.
- ctime: the time, the metadata entry of a file itself was last updated.

2.2 Levenshtein Distance

The Levenshtein distance is a string metric introduced by Vladimir Levenshtein [Lev66]. For two given strings the Levenshtein distance is indicating the minimum amount of edit operations required to transform the first string into the second. Thereby, the possible edit operations are insertion, deletion or substitution of a single character in the string. As a simple example, we want to calculate the Levenshtein distance between the the two strings `a = 'house'` and `b = 'spouse'`. Since the string length of string `a` is 4 and of string `b` is 5, we obviously need to insert one character in string `a`. Thus, the Levenshtein distance is at least 1. So, if we insert an `s` at the beginning of string `a` and then substitute the `h` with an `p`, we successfully transformed string `a` into string `b`. For this transformation, we required at least an insertion and a substitution operation and therefore the Levenshtein distance between `'house'` and `'spouse'` is 2.

2.3 Clustering with DBSCAN

Ester et al. [EKSX96] created the well-known and broadly used data clustering algorithm *DBSCAN*. Figure 1 depicts the concepts of *density-reachability* and *density-connectedness* in DBSCAN. DBSCAN requires the following two parameters:

- `eps`: the maximum distance between two objects
- `minPts`: the minimum number of objects required to create a cluster

Two objects are considered to be directly density-reachable if the distance between both objects in not greater than the distance defined by `eps` and there is a sufficient number (`minPts`) of other objects in the cluster. Object `m` is thus directly density-reachable from object `p` for `minPts = 4` in our example. Object `q` is not directly density-reachable from object `p` but is from object `m`, but still `q` is considered to be density-reachable from `p`, since there is a link of directly density-reachable objects from `p` to `q`. Objects like `s` or `t` on the edge of a cluster are considered to be density-connected since they both are density-reachable from object `q`, even if there is not a sufficient number (`minPts`) of objects within

their maximum distance (`eps`). A cluster of objects in DBSCAN consists of those objects that are all density-connected.

After we discussed the basics of our approach, the following section covers the details of *zero-knowledge event reconstruction*.

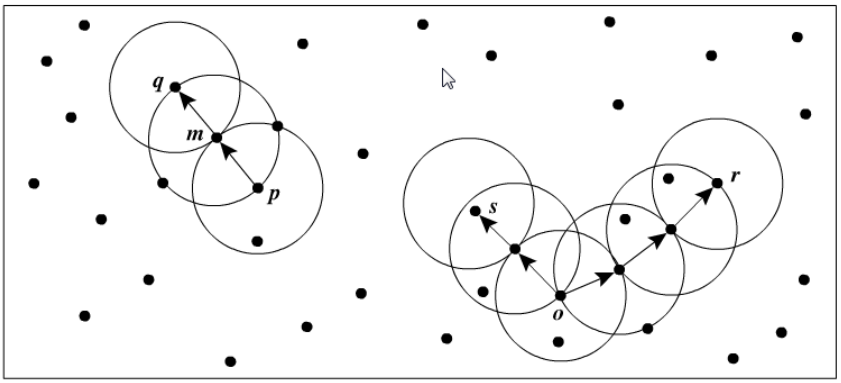


Figure 1: *density-reachability* and *density-connectedness* in DBSCAN [Han12]

3 Zero-Knowledge Event Reconstruction

As discussed previously, automatic event reconstruction approaches either require an up front, time consuming profiling phase or an individually written parser or analyzer so far. Either way, prior knowledge of application specifics such as file storage location, log file format or timestamp modification patterns are required.

Our approach is to spare prior knowledge on the application level, by utilizing the filesystem's timestamp information update behavior on a significant amount of files for each user triggered activity. Grier [Gri11] observed that under normal usage filesystem activity is neither uniformly nor normally distributed over files but is following a Pareto distribution. Thus, we argue that for individual actions, their execution lead to timestamp metadata updates on a significant amount of files stored in the filesystem. For example, when starting an application, not only the corresponding binary file is accessed but many other files such as linked libraries, configuration files, images, media files or log files are accessed or even modified or created. In case of such an event, the filesystem then updates the timestamp information accordingly for each file. Since those file accesses happen close to each other in terms of time, their corresponding timestamp values do not differ by more than a few seconds from one another.

The idea is to extract all timestamp information and the corresponding file name including the absolute path from a filesystem of interest and sort them by their timestamp values in ascending order. This basically results in a timeline of all file creations, file modifications and file accesses of the entire volume. Since different applications and events mostly access different files and because of the Pareto distribution, peaks (in terms of amount of

files) arise on the timeline. In other words, each peak corresponds at least to one event that caused these different timestamp updates. Therefore and for simplicity, we refer to such peaks as events in the next sections.

Each event consists of different files that were accessed, modified or created during the same time. Therefore, files that belong to the same application and are located in the same locations (e.g. in the same subdirectory) are clustered. Since, application specific files may additionally be distributed over different locations in the filesystem, multiple clusters might get created for each event. For instance, application data is most of the time stored apart from operating system libraries, temporary files, or application configuration data.

Moreover, the type of timestamp update is being analyzed further, because this can give an explanation about the actual event and the modification of files. Based on these clusters and timestamp analyses, a label is created for each event, indicating which application was used or which files were accessed or modified. These resulting events can then be visualized in a timeline fashioned style, to support the investigator in the event reconstruction phase, by providing a quick and easy overview of all identified events.

4 Prototype Implementation

To demonstrate our approach of *zero-knowledge event reconstruction* we implemented a prototype called `0KER.py` (0-Knowledge Event Reconstruction) in Python. In the following sections, we have a closer look at the details of the three main stages of `0KER.py`: the detection of timestamp structures, event labeling and the timeline visualization.

4.1 Detection of Events in Timestamp Structures

`0KER.py` operates on XML representations of filesystem metadata generated by *fiwalk* [Gar09]. In a first step, the *fiwalk* XML report is parsed and all file names including their path and timestamp information is extracted. Then, in a second step, we sort this data by their timestamp values in ascending order, before the events then are determined by aggregating those files that have been accessed, modified or created sequently. Whenever a gap between two consecutive timestamp values of more than 20 seconds is detected, a new event is being created. Our evaluation has shown, that 20 seconds was, at least in our case, the value leading to optimal results. On 20 different disk images we observed, that for values smaller than 20 seconds the amount of events that were mistakenly divided into two or more events rises. While, for values greater than 20 seconds, the amount of events that mistakenly contain traces of two or more events increases. Moreover, this approach has shown to be more suitable for event detection than clustering algorithms like k-means.

In a pre-evaluation, we have observed, that clustering algorithms that are based on partitioning are not well suited for automatic event reconstruction, since k-means, k-medoids or other alike algorithms aim to cluster `n` objects in `k` clusters. Even though `k` is variable among multiple runs, a fixed value needs to be chosen before each clustering run. Since the

amount of events is unknown to the investigator prior to the investigation, clustering based on partitioning is not applicable. On the other hand, due to the sparseness of timestamp values and the Pareto distribution observed by Grier [Gri11], the approach of identifying events based on the distance of timestamp values among each other, has shown to be well suited.

4.2 Event Labeling

The labeling of events is done by analyzing the file types in combination with the timestamp type of all files within each event. If a pattern of timestamp types, according to the rules depicted in Figure 3 is found, the corresponding event description is assigned to the appropriate file. This way, newly created files, copied files as well as files that have been modified during a specific event are identified and reported to the investigator. Additionally, known file types like executables, documents, media files, audio files and databases are analyzed separately. For example, if a file access on an executable file is detected, the event is labeled as "Application `app.exe` started". If two or more events on a system happened around the same time, the chance is that those events are regarded as a single event due to the event detection mechanism described above. In such cases, an event contains for example multiple application startups. Therefore, we still list all generated labels as a result, so that the investigator is able to notice both events.

Moreover, the timestamp changes within each event are clustered according to their file names and paths. This is done, in order to group all files in the same storage locations. The following section describes the clustering in more detail.

4.2.1 Path & Filename Clustering

The clustering is done in two steps. In the first step, we apply the concept of string metrics by calculating the Levenshtein distances for the string representations of the absolute path (including the filenames) of all files that have been accessed around the same point in time. Thereby, smaller values indicate files that are located in the same subdirectories or share similar folder structures. Thus,we are able to create a distance matrix which is, in the second step, used as input for the DBSCAN algorithm. As a result of DBSCAN, all files that have a small Levenshtein distance and thus share the same subdirectories for example, are clustered. These clusters ease the analyzing process for investigators and reduce the amount of data to be sighted, by consolidating multiple timestamp updates that occurred during the same time and belong to the same application.

For example, the start of an application causes program data located in `\Program Files \application\` to be accessed. Additionally, shared libraries are loaded from `\Windows \system32\` and configuration files stored in `\Users\username\AppData\` are read. In this example, the following three clusters would be identified by DBSCAN for this event. One cluster containing all files accessed in the applications installation folder. The other cluster contains the shared libraries of the Windows installation folder and a third cluster is

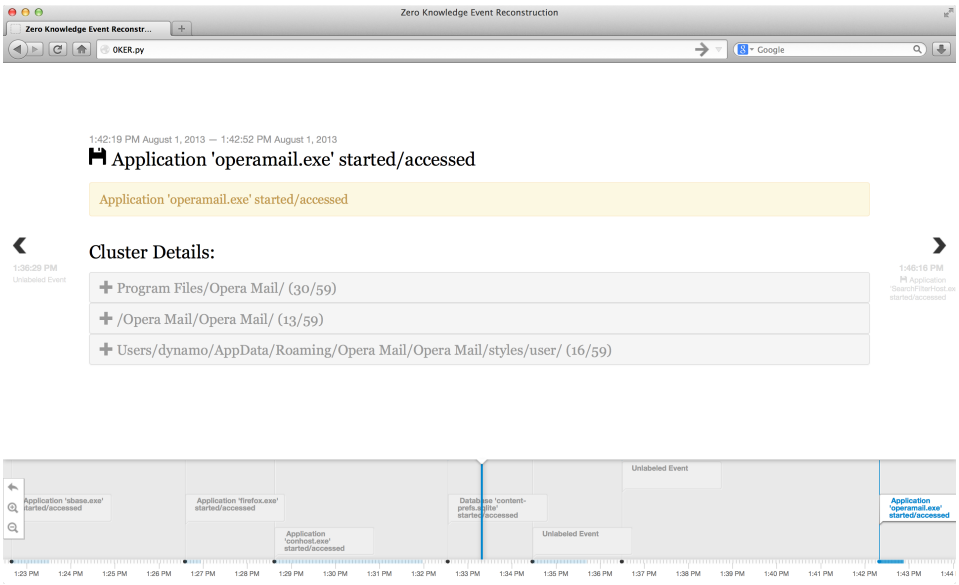


Figure 2: Visualization of events found with 0KER.py

created for the user specific application data stored in the user's home folder. Additionally, for each cluster the amount of files that have been accessed, created or modified is counted. Using this information, the investigator is able to quickly identify the important clusters for each event. Moreover, two events that happened simultaneously and thus were regarded as a single event, are easier to distinguish by investigators since files from one application are grouped in other clusters than files from another application.

As described in Section 2.1, DBSCAN requires two parameters, namely *minPts* and *eps*. Where *minPts* determines the minimal amount of objects required to form a cluster and *eps* indicates the maximum distance between two objects in the same cluster. More precisely, in our case *eps* is set to the maximal distance, measured as Levenshtein distance, the absolute path (including the filename) of two files may have in order to be contained in the same cluster. *minPts*, on the other hand, indicates the amount of timestamp updates of different files that are required in order to create a new cluster. In total, we evaluated 12 different combination sets of *eps* and *minPts* on 6 different images. In our case, the optimal value for *minPts* is 12 and 50 for *eps*. These values led to the best results in terms of detection rates, amount of events and correctly labeled events.

Additionally to the clustering of files, we further analyze the type of timestamp modification. We highlight files that were created, updated, copied from another location or renamed by checking if the timestamp values found on disk match a known pattern as depicted in Figure 3. This is done to provide the investigator a quick overview of how many files were accessed, which files were newly created or which files have been modified during this event and where they are located. The amount of files accessed might help the investigator to verify the automatic identification of this event as well as to distinguish two independent events that were mistakenly identified as one single event.

4.3 Visualization

Timelines are intended to support the investigator in early stages of the event reconstruction phase of a digital investigation. Our goal is to provide a quick and easy overview of all found and identified events that may point the investigator to suspicious files, applications, system behavior or other activities for further in-depth analysis.

Figure 2 depicts a timeline created with `0KER.py` and visualized by TimelineJS [Ger13]. TimelineJS is an open source, javascript, json and html based timeline visualization tool created by the Northwestern University Knight Lab. Each event is marked in a balloon-shaped box in the timeline view located at the bottom of the page. The yellow-colored info box holds the identified label for the activity. In case multiple labels exist for an event, all labels are listed, to indicate the investigator that possibly two or more activities happened at that point in time. For each identified cluster of files a grey-colored, collapsible box is created, that is labeled with the longest common substring of all path names within the cluster. Upon expanding a cluster, a list of all files and subfolders that were accessed, modified, or created around the same time is shown. With this detailed information, the investigator is able to understand and verify the results of the automatic event reconstruction done by `0KER.py`.

In the following section, we are now going to evaluate our implementation of `0KER.py` and compare our approach to the results of the fingerprinting framework *Py3xF*.

5 Evaluation

For our evaluation, we installed Microsoft Windows 7 along with 10 different applications on two independent virtual machine disk images. Oracle's VirtualBox was used as virtualization platform and the installed applications are: *Mozilla Firefox, Mozilla Thunderbird, Opera Browser, Opera Mail, Google Chrome, Pegasus Mail, OpenOffice, LibreOffice, ICQ* and *WinSCP*. On each machine, we then automatically and consecutively started 5 different applications and denoted the time of execution. After powering off the machines, we extracted the file system's timestamp metadata from both images using *fiwalk*. Next, we started `0KER.py`. As a result, for the first disk image, 7 events and for the second image 8 events were detected and reported. Table 1 and Table 2 show which actions were found with `0KER.py` in comparison with *Py3xF*. As can be seen in Table 1, `0KER.py` correctly identified, matched and labeled the startup process of *LibreOffice Base, Mozilla Firefox* and *Opera Mail*, which is indicated by the checkmark (✓). The startups of *ICQ* and *Google Chrome* on the other hand, were marked with an **o**, since those events could not be labeled correctly. This is the case, whenever the accessed timestamp of the according application executable is updated in later points of time for example. However, these events were not marked as unidentifiable (✗), since the report contains at least one event entry for both startup processes with significant clusters of timestamp updates for the corresponding application, indicating that the application was in fact running during that point in time. *Py3xF* was able to correctly identify and label all but the Mozilla Firefox startup process,

based on fingerprints that have previously been generated for these 10 applications. In this case, *Py3xF* reported an action performed with Mozilla Thunderbird instead.

On the second image, we not only performed the startup process for the individual applications, but we also send out an email using Mozilla Thunderbird. As can be seen in Table 2, *Py3xF* was able to correctly identify and label all performed application startups and the event of sending an email with Thunderbird. `0KER.py` on the other hand, failed to identify the start of WinSCP. The event of sending an email was reported in three different events and may be reconstructed by the investigator. Two events were created stating, that the database files `addons.sqlite`, `extensions.sqlite` and `places.sqlite` of Thunderbird were modified. But more importantly, an event was created that indicated the access and modification of the IMAP store of the corresponding Thunderbird email profile in `AppData/Roaming/Thunderbird/Profiles/fsk381mz.default/ImapMail/imap.googlemail.com/[Gmail].sbd/`.

As mentioned earlier, although we only performed 5 actions on each disk , 7 events were reported for the first image and 8 for the second one. This is because on the first image the events for starting Mozilla Firefox and Google Chrome were split up in two events each. On the second image the additional events were created for the event of sending an email in Mozilla Thunderbird.

As we can see, with our zero-knowledge event reconstruction approach, it is possible to automatically identify and label most application startup events correctly.

Event	0KER.py	Py3xF
LibreOffice Base	✓	✓
Mozilla Firefox	✓	✗
ICQ	**o**	✓
Google Chrome	**o**	✓
Opera Mail	✓	✓

Table 1: Results of disk image 1

Event	0KER.py	Py3xF
Thunderbird/send mail	✓/**o**	✓/✓
Pegasus Mail	✓	✓
OpenOffice Calc	✓	✓
WinSCP	✗	✓
Opera Browser	**o**	✓

Table 2: Results of disk image 2

However, application specific activities such as sending an email can only be matched by tools based on prior knowledge of the application or have to be identified by the investigator. With the concept of clustering timestamp updates around certain points in time, based on their file name and path, we introduced a possibility to reduce the amount of data to be analyzed by investigators in order to identify such application specific activities.

6 Conclusion

In this paper, we have presented an approach to event reconstruction in post-mortem investigations that does not depend on application specific parsers or fingerprints. A prototype implementation called `0KER.py` was developed for Microsoft Windows systems based on the NTFS filesystem.

With our implementation, which identifies timestamp update aggregations around certain points in time, we were able to reconstruct particular user activities and highlight events that occurred but could not be tied to a specific application. With this approach, it is possible to reconstruct events such as application installations and startups, as well as access and modification of application specific files like configuration files, temporary files or data stores. Whereas the identification of application specific user activities such as sending or receiving an email, that may be automatically identified by approaches like *Py3xF*, still requires expert knowledge of investigators.

6.1 Limitations

Obviously, the timeline generated by `0KER.py` only contains the latest occurrences of different events, since NTFS only stores the latest timestamp values in the $MFT and there is no history of previous values. Additionally, in cases, when two or more events happen simultaneously or right after another, those might be mistakenly considered and shown as one single event in the timeline. This is due to the nature of the basic approach of event detection discussed in Section 4.1. Moreover, compared to application specific approaches like *log2timeline* or *Py3xF*, that are able to also identify individual user activities with applications, `0KER.py` might only label events for application startup or installation as well as document and database access and modifications. Nevertheless, by clustering, `0KER.py` greatly reduces the amount of events that has to be sighted by an investigator and highlights timestamp update aggressions on the timeline.

References

[Car05] Brian Carrier. *File System Forensic Analysis*. Addison-Wesley Professional, 2005.

[Cas11] Eoghan Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet*. Academic Press, third edition, 2011.

[EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[Gar09] S. Garfinkel. Automating disk forensic processing with SleuthKit, XML and Python. In *Proceedings of the 2009 Fourth International IEEE Workshop on Systematic Approaches to Digital Forensic Engineering*, pages 73–84, 2009.

[Gar10] Simson Garfinkel. Digital Forensics Research: The Next 10 Years. *Digital Investigation*, 7:64–73, 2010.

[Ger13] Joe Germuska. Timeline JS - Beautifully crafted timelines that are easy, and intuitive to use. `http://timeline.knightlab.com/`, November 2013.

[Gri11] Jonathan Grier. Detecting data theft using stochastic forensics. *Digital Investigation*, 8:S71–S77, August 2011.

[Gud13] Kristinn Gudjonsson. log2timeline `https://code.google.com/p/log2timeline/`, November 2013.

[Han12] Jiawei Han. *Data mining : concepts and techniques*. Morgan Kaufmann series in data management systems. Morgan Kaufmann/Elsevier, Waltham, MA, 3rd ed edition, 2012.

[HP12] Christopher Hargreaves and Jonathan Patterson. An automated timeline reconstruction approach for digital forensic investigations. *Digital Investigation*, 9, Supplement(0):S69 – S79, 2012. The Proceedings of the Twelfth Annual {DFRWS} Conference.

[JGZ11] Joshua Isaac James, Pavel Gladyshev, and Yuandong Zhu. Signature Based Detection of User Events for Post-mortem Forensic Analysis. In *Digital Forensics and Cyber Crime*, volume 53 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 96–109. Springer Berlin Heidelberg, 2011.

[KDF13] Sven Kälber, Andreas Dewald, and Felix C. Freiling. Forensic Application-Fingerprinting Based on File System Metadata. In *IT Security Incident Management and IT Forensics (IMF), 2013 Seventh International Conference on*, pages 98–112, 2013.

[Lee10] Rob Lee. Windows 7 MFT Entry Timestamp Properties `http://computer-forensics.sans.org/blog/2010/04/12/windows-7-mft-entry-timestamp-properties`, April 2010.

[Lev66] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.

[SDF13] Johannes Stüttgen, Andreas Dewald, and Felix C. Freiling. Selective Imaging Revisited. In *7th International Conference on IT Security Incident Management & IT Forensics (IMF)*. IEEE, 2013.

[Tur06] P. Turner. Selective and intelligent imaging using digital evidence bags. *Digital Investigation*, 3:59–64, 2006.

Appendix

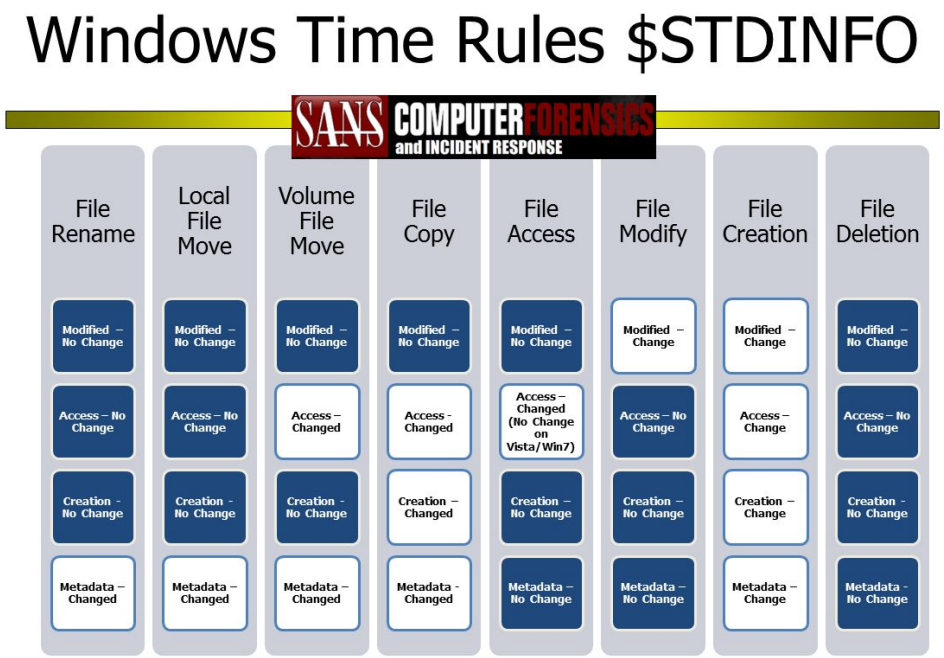


Figure 3: Windows Time Rules [Lee10]

Komplexe Systeme, heterogene Angreifer und vielfältige Abwehrmechanismen: Simulationsbasierte Entscheidungsunterstützung im IT-Sicherheitsmanagement*

Andreas Ekelhart, Bernhard Grill, Elmar Kiesling
SBA Research
Favoritenstraße 16
1040 Wien, Österreich
{aekelhart,bgrill,ekiesling}@sba-research.org

Christine Strauß
Universität Wien
Oskar-Morgenstern-Platz 1
1090 Wien, Österreich
christine.strauss@univie.ac.at

Christian Stummer
Universität Bielefeld
Universitätsstraße 25
33615 Bielefeld, Deutschland
christian.stummer@uni-bielefeld.de

Abstract: Dieser Beitrag beschreibt einen simulationsbasierten Entscheidungsunterstützungsansatz zur Analyse und Optimierung der Sicherheit komplexer Informationssysteme. Er stützt sich auf die konzeptuelle Modellierung von Sicherheitswissen, die Modellierung des Angreiferverhaltens, die Simulation von Angriffen sowie auf genetische Algorithmen zur Bestimmung effizienter Bündel von Sicherheitsmaßnahmen. Mittels Angriffssimulationen für unterschiedliche interne und externe Angreifertypen können Vertraulichkeits-, Integritäts- und Verfügbarkeitsrisiken erfasst und für die Ermittlung effizienter Kombinationen von Sicherheitsmaßnahmen hinsichtlich mehrerer Risiko-, Kosten- und Nutzen-Ziele genutzt werden. Wir beschreiben den entwickelten Ansatz sowie seine prototypische Implementierung und zeigen die Anwendung anhand von beispielhaften Szenarien.

1 Einführung

Der Schutz komplexer Informationssysteme stellt heute, nicht zuletzt angesichts der dynamischen und sich rasch verändernden Bedrohungslandschaft, eine erhebliche Herausforderung dar. In den vergangenen Jahren war eine deutliche Tendenz zu komplexen und zielgerichteten Angriffen beobachtbar. Im Gegensatz zu Schadsoftware, die automatisiert vordefinierte technische Schwachstellen ausnutzt, richten sich diese Angriffe gegen spezifisch und strategisch gewählte Ziele, die häufig über komplexe Angriffsketten erreicht werden. Die ausschließlich isolierte Betrachtung einzelner technischer Maßnahmen ist daher unzureichend und aus Sicht eines systematischen Risikomanagements wenig zielführend. Viel-

*Ein ähnlicher Beitrag erschien in englischer Sprache [KEG+14].

mehr sollte eine umfassende Betrachtungsweise verfolgt werden um adäquat berücksichtigen zu können, dass Angreifer zur Zielerreichung technische wie auch nicht-technische Mittel einsetzen und kombinieren (z.B. Netzwerk, Software, physische, soziale Angriffe). Der hier vorgestellte Ansatz unterstützt das IT-Sicherheitsmanagement durch eine dynamische Angriffssimulation bei der Bewertung der Sicherheit von Systemen sowie bei der Bestimmung effektiver Kombinationen von Sicherheitsmaßnahmen. Wir betrachten Sicherheit dabei als emergente Eigenschaft komplexer Informationssysteme.

Basierend auf einer Reihe von Formalismen und Methoden (Monte Carlo, ereignisorientierte Simulation, Petrinetze etc.) wurden in der Literatur unterschiedliche Ansätze zur Sicherheitsanalyse mittels simulierter Angriffe vorgestellt [Coh99, CPJL01, DW06, DMCR06]. Diese modellieren jedoch den Angreifer und sein Verhalten in der Regel nicht explizit. Im Kontext der Identifikation möglicher Angriffe setzt unser Ansatz auf dynamische Angriffsgraphen. Im Unterschied zu bestehenden Arbeiten [AWK02, OBM06, SO08] verfolgen wir aber nicht das – in komplexeren Umgebungen häufig nicht erreichbare – Ziel einer vollständigen Enumeration aller Pfade, sondern ermitteln Angriffsgraphen dynamisch im Zuge der Simulation von Angriffen, was es uns auch erlaubt, auf einschränkende Annahmen, wie etwa Monotonie (d.h. dass einmal erlangte Angriffsmöglichkeiten nicht – etwa durch Abstürzen eines Rechners aufgrund eines versuchten Angriffes – wieder verloren werden können), zu verzichten. Weiters beschränken wir uns in unserer Modellierung nicht auf den Aspekt der Netzwerksicherheit, sondern erfassen Wissen über mögliche physische, soziale, und technische Angriffe und deren Kausalzusammenhänge in einer formalen Wissensbasis. Dabei greifen wir auf bestehende Konzepte einer in [FE09] vorgestellten Ontologie zurück. Um die im Simulationsablauf häufigen Abfrage der Wissenbasis performant durchführen zu können erfolgt die Modellierung der Wissensbasis jedoch ähnlich wie in [OBM06] in Form von Prolog-Regeln. In Bezug auf Mehrzieloptimierung ist der vorgestellte Ansatz mit [EFN09] verwandt, hebt sich aber durch den simulationsbasierte Bewertung und die Berücksichtigung mehrstufiger Angriffe ab.

Die zentralen Forschungsbeiträge können wie folgt zusammengefasst werden: Wir modellieren abstrakte Angriffsmuster und entwickeln einen Mechanismus, um daraus konkrete Angriffspfade abzuleiten. Während bestehende Ansätze häufig darauf abzielen die Menge aller möglichen Angriffspfade aufzuzählen betrachten wir die Bestimmung dieser Pfade als einen dynamischen, durch das Verhalten des Angreifers gesteuerten Prozess. Wir modellieren Angreifer daher als Agenten, die sich strategisch verhalten und bewusste Entscheidungen treffen. Schließlich entwickeln wir eine ereignisorientierte Angriffssimulation und nutzen diese, um mit Hilfe genetischer Algorithmen unter mehrfacher Zielsetzung effektive Kombinationen (Portfolios) von Sicherheitsmaßnahmen zu bestimmen. Abbildung 1 stellt die Architektur des simulationsbasierten Optimierungsansatzes dar. Dieser baut auf einer Wissensbasis auf, welche Angriffsmechanismen, Gegenmaßnahmen und das zu schützende System formal abbildet (Kapitel 2). Auf Basis dieser Elemente können Angriffsmuster dynamisch verbunden und Angriffe simuliert werden (Kapitel 3). Die automatisierte Bestimmung effizienter Kombinationen von Sicherheitmaßnahmen erfolgt schließlich durch metaheuristische Optimierungstechniken (Kapitel 4). Entscheidungsträger können auf Basis der Optimierungsergebnisse effiziente Lösungen vergleichen, mehrere zueinander in Konflikt stehende Ziele abwägen und letzlich eine Kombina-

tion von zu implementierenden Sicherheitsmaßnahmen auswählen. Wir verdeutlichen die Anwendung der Methode anhand eines Beispiels (Kapitel 5).

2 Wissensbasis

Eine Wissensbasis, welche die benötigten Konzepte und Zusammenhänge aus der Sicherheitsdomäne erfasst, dient als Grundlage für die in Abschnitt 3 vorgestellte Angriffssimulation. Unsere Implementierung verwendet die deklarative, logische Programmiersprache *Prolog* zur Modellierung des sicherheits- und systemrelevanten Wissens sowie zur Erkennung möglicher Angriffspfade.

2.1 Sicherheitswissen

Das *attack and control model* liefert eine formale Beschreibung, wie Systeme angegriffen und abgesichert werden können. Es beschreibt Angriffsmuster, die sich auf einen bestimmten Kontext beziehen und nur unter bestimmten Voraussetzungen gültig sind. Wir nutzen das bestehende, öffentlich verfügbare CAPEC-Repository[1] (common attack pattern enumeration and classification), welches aktuell 400 Angriffsmuster aus dem Softwarebereich umfasst, die Sicherheitswissen aus Angreifersicht repräsentieren.

Um diese zumeist semi-strukturierte Information für unseren Simulationsansatz nutzen zu können, ist eine Übersetzung in eine formale Darstellung (auf Basis der CAPEC Abschnitte *Summary*, *Experiments*, *Outcomes* und *Attack Prerequisites*) erforderlich. Das folgende Beispiel beschreibt eine *SQL Injection* mit den nötigen Vorbedingungen. Das Muster liefert mittels der hinterlegten Datenbasis alle zulässigen Variablenbelegungen für *Attacker* und *DbServer*.

```
action_sqlInjection(Attacker, DbServer) :-
        technicalSkillLevel(Attacker, TechnicalSkillLevel),
```

[1] http://capec.mitre.org/, letzter Zugriff am 13. Februar 2014.

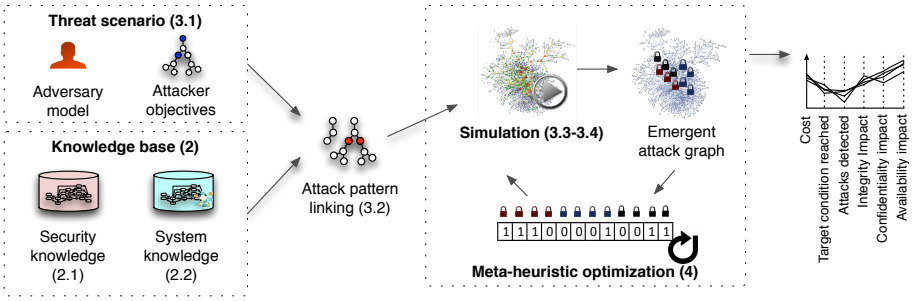


Abbildung 1: Architektur des simulationsbasierten Optimierungsansatzes

```
        TechnicalSkillLevel >= 1,
        connected(AttackHost, WebServer, httpProtocol, httpPort),
        connected(WebServer, DbServer, dbProtocol, dbPort),
        ...
```

Um mögliche Auswirkungen einer Angriffsaktion abzubilden, definieren wir Nachbedingungen, welche die in der Simulation auftretenden Zustandsveränderungen formalisieren. Die erfolgreiche Ausführung einer Angriffsaktion kann dem Angreifer beispielsweise Root-Rechte auf dem Zielserver sichern, während hingegen ein fehlgeschlagener Versuch den attackierten Server offline nehmen könnte. Formal definieren wir beide Typen als Regeln, welche eine Änderung in der Wissensbasis bewirken.

In CAPEC findet sich auch ein Abschnitt *CIA Impact*, welcher die Auswirkungen einer Angriffsaktion hinsichtlich Vertraulichkeit, Integrität und Verfügbarkeit auf einer Skala von *high*, *medium* und *low* bewertet. Obwohl diese Information einen nützlichen Anhaltspunkt hinsichtlich einer Bewertung liefert, wird auf den Kontext, in dem der Angriff erfolgt, keine Rücksicht genommen. Unser simulationsbasierter Ansatz hingegen trägt dem Umstand Rechnung, dass gleiche Angriffsaktionen, je nach Angriffsziel und Umgebung, unterschiedliche Auswirkungen haben können. Die Sicherheitswissensbasis definiert nur die betroffenen Sicherheitsattribute einer Angriffsaktion sowie Regeln, wie das Schadensausmaß erhoben wird. Das folgende Beispiel legt fest, dass sich *SQL Injection* auf das Sicherheitsattribut *Vertraulichkeit* auswirkt. Gelingt es einem Angreifer die Aktion erfolgreich durchzuführen, wird die Auswirkung durch die Vertraulichkeitsbewertung des betroffenen Servers bestimmt.

```
action_impact(action_sqlInjection, confidentiality).
impact_success_sqlInjection(Attacker, TargetHost,  Impact):-
        importance(TargetHost, confidentiality, Impact).
```

Zusätzlich erweitern wir das Sicherheitswissen um Maßnahmendefinitionen aus CAPEC. Die Abschnitte *Solutions*, *Mitigations* und *Relevant security requirements* liefern hierzu die Grundlage.

2.2 Systemwissen

Um den Sicherheitszustand eines Informationssystems bestimmen zu können, müssen dessen Komponenten erhoben und Zusammenhänge modelliert werden. Zu diesem Zweck erfassen wir – getrennt von dem bereits erläutertem abstrakten Angriffswissen – systemspezifisches Wissen. In diesem Modell werden materielle sowie immaterielle *Assets* wie z.B. Hardwarekomponenten, Netzwerkstrukturen, Daten und Mitarbeiter abgebildet. Das folgende Beispiel einer Intrusion Detection Maßnahme zeigt, wie *Assets* modelliert und Maßnahmen zugewiesen werden.

```
host(dbServers_host_1).
installed(dbServers_host_1, ids1).
stores(dbServers_host_1, projectDb1).
inHostGroup(dbServers_host_1, dbServerHosts).
hacl(workstationHosts, dbServerHosts, httpProtocol, httpPort).
```

Zur späteren Bewertung der Auswirkung von Bedrohungen werden modellierten Assets Kritikalitätswerte für Sicherheitsattribute zugewiesen. Diese Werte können aus existierenden Impact-Analysen oder Asset-Kritikalitätsanalysen übernommen werden.

```
criticality(projectDb1, confidentiality,2).
criticality(projectDb1, integrity, 3).
criticality(projectDb1, availability, 2).
```

3 Angriffssimulation

3.1 Bedrohungsprofil

In der Literatur werden Angreifer üblicherweise anhand natürlichsprachlicher Beschreibungen charakterisiert und klassifiziert [MRKS10]. Ein Vorteil unseres formalen Modells besteht darin, dass es die Definition spezifischer Profile unter Berücksichtigung von Fähigkeiten, Ressourcen und Verhaltensattributen des Angreifers ermöglicht. Diese Attribute bestimmen, welche Angriffsmöglichkeiten für einen Angreifer verfügbar sind, das Angriffsverhalten, sowie die Erfolgswahrscheinlichkeiten für Angriffsaktionen.

Die Angriffssimulation nutzt und verbindet das modellierte Wissen für die Ausführung von Angriffsszenarien. Die Entscheidungen der Angreifer für bestimmte Aktionsfolgen, das Ergebnis einzelner Aktionen sowie die Erkennung von Angriffen werden in der Simulation probabilistisch ermittelt. Um Variabilität und Unsicherheit abzubilden, werden für jedes Angriffsszenario mehrere Replikationen mit unterschiedlichen Startwerten des Zufallszahlengenerators ausgeführt.

Bei jedem Durchgang arbeitet die Simulation eine Liste von Ereignissen ab und sammelt Ergebnisvariablen (z.B. impacts auf Sicherheitsattribute), die aggregiert und zu Optimierungszwecken genutzt werden können. *Action Selection* Ereignisse ermitteln durch Verbindung von Angriffsmustern verfügbare Aktionen, wählen auf Basis des Angreifer-Verhaltensmodells Aktionen aus und fügen *Action Start* Ereignisse in den Simulationsablauf ein. Während der Ausführung wird die effektive Dauer dieser Aktionen (entsprechend der Schwierigkeit, Angreiferfähigkeiten und eingesetzten Präventivmaßnahmen) bestimmt und ein *Action End* Ereignis in den Simulationsablauf eingefügt. *Detection* werden werden ausgelöst, sobald Assets angegriffen werden, die mit detektierenden Maßnahmen versehen sind; anschließend kann ggf. ein *Attacker Stopped* Ereignis ausgelöst werden. Schließlich beendet ein *Target Condition Reached* Ereignis die Simulation, falls der Angreifer sein Ziel erreicht hat.

3.2 Verkettung von Angriffsmustern

Angriffsmuster werden automatisch aufgrund gemeinsamer Vor- und Nachbedingungen verbunden. Dieser Ansatz ist mit Angriffsgraph-Konzepten [Sch00] verwandt, insbeson-

dere mit fortgeschrittenen Formalismen und Methoden zur Generierung großer Bäume [AWK02, SHJ+02, SO08] sowie mit Erweiterungen, die Sicherheitsmaßnahmen einbeziehen [BFP06]. Allerdings beruhen diese Ansätze darauf, *alle* möglichen Angriffspfade zu identifizieren, um einen vollständigen Angriffsgraph zu erhalten. Dies ist aufgrund der exponentiellen Größe des Suchraums rechnerisch problematisch. Unser Ansatz begreift die Suche nach möglichen Angriffspfaden hingegen als dynamischen Prozess und ist daher in vieler Hinsicht allgemeiner und flexibler.

3.3 Auswahl und Ausführung von Aktionen

Wir implementieren Angreifer mit einem Verhaltensmodell, das Aktionen iterativ aufgrund von (i) individuellen Charakteristika, (ii) dem allgemeinen Wissen des Angreifers über mögliche Angriffspfade und (iii) dem Ergebnis vorausgegangener Angriffsaktionen auswählt. Dieses Modell berücksichtigt, dass Angreifer (in unterschiedlichem Ausmaß) über allgemeines Sicherheitswissen verfügen, aber typischerweise nur unvollständige Informationen über das angegriffene System haben. Das allgemeine Wissen über mögliche Angriffspfade wird durch einen abstrakten Angriffsgraph repräsentiert, der aus der Wissensbasis für einen bestimmten Angreifer und ein betrachtetes Ziel abgeleitet werden kann. Die Simulation berücksichtigt die gültigen Zuordnungen für alle Vorbedingungen einer abstrakten Aktion im Kontext eines modellierten Systems und generiert daraus eine Menge konkreter Instanzen von Angriffsaktionen, die gegen bestimmte Assets ausgeführt werden können.

Der im Anhang dargestellte Verhaltensalgorithmus beruht auf der Annahme, dass Angreifer wechselweise einer gewählten Strategie folgen (d.h. eine Sequenz von logisch zusammengehörigen Aktionen ausführen) und alternative Ansätze versuchen (d.h. Startpunkte für einen neuerlichen Angriffsversuch wählen). Wir unterstellen ferner, dass Angreifer den erwarteten Aufwand minimieren und daher Angriffe im abstrakten Graph möglichst "nahe" am Zielzustand beginnen. Diese Annahme wird in der Funktion $choose(\overline{A})$ umgesetzt, welche die relativen "Distanzen" (Anzahl von Schritten in Bezug auf die längste Distanz) zwischen einer abstrakten Aktion a und der Zielbedingung t im abstrakten Angriffsgraph für alle möglichen Aktionen berechnet. Schließlich ermittelt die Funktion Gewichtungen für jede mögliche konkrete Aktion $\overline{a}$ auf Basis der Erfolgs- und Detektionswahrscheinlichkeiten sowie der ermittelten relativen Distanzen zur Zielbedingung und der Präferenzen des Angreifers. Die Funktion $weightedChoice(\overline{A}, W)$ bestimmt anschließend die gewählte konkrete Aktion mit entsprechenden Wahrscheinlichkeiten.

Nachdem die erste Aktion ausgewählt wurde, sind alle weiteren Entscheidungen vom Ergebnis der zuvor gewählten Aktion p abhängig. Die Verhaltensparameter $p_{continueNew}$, $p_{alternativesNoNew}$, p_{retry}, $p_{AlternativesFailed}$ bestimmen dabei abhängig vom Ergebnis, ob der Angreifer (i) eine neu verfügbar gewordene Aktion, (ii) eine alternative Aktion auf gleicher Ebene (d.h., eine Aktion, die als Konsequenz der gleichen vorhergehenden Aktion verfügbar wurde) oder (iii) einen neuen Ansatz versucht und aus allen verfügbaren Aktionen wählt.

Nach Ausführung einer Aktion wird bestimmt, (i) ob die Angriffsaktion erfolgreich ist, (ii) ob die Zielbedingung erreicht wird, (iii) welche Aktionen als Konsequenz nicht mehr verfügbar sind, (iv) welche neuen Aktionen verfügbar werden und (v) welche Auswirkungen auf Sicherheitsattribute entstehen.

3.4 Auswirkung detektierender Maßnahmen

Für detektierende Maßnahmen können zwei mögliche Konsequenzen definiert werden, nämlich (i) *stop*, d.h. Simulationslauf beenden, oder (ii) einen Verweis auf einen definierten Nachbedingungspfad im abstrakten Graph, der ausgeführt werden soll. Letzteres kann Zustände im Systemmodell verändern. Wenn beispielsweise ein Intrusion Detection System die Quell-IP-Adresse des Angreifers blockiert, so verhindert das bestimmte Angriffe, ermöglicht aber potentiell auch neue Angriffsmuster (z. B. wenn dies genutzt werden kann, um legitime Zugriffe zu unterbinden).

4 Optimierung

Entscheidungsträger können durch Modellierung ihrer Systeme, Einsatz von Sicherheitsmaßnahmen und Simulation von Angriffen wertvolle Erkenntnisse über die Sicherheit eines Systems gewinnen. Manuelles Experimentieren mit Maßnahmen, bis zufriedenstellende Simulationsergebnisse erzielt werden, ist jedoch ein aufwändiger Prozess, der bei komplexeren Systemen praktisch nicht sinnvoll durchführbar ist.

Wir führen daher das Konzept von Sicherheitsmaßnahmen-Portfolios ein, die durch einen Genotyp beschrieben und automatisch optimiert werden. Jeder Genotyp setzt sich aus binären Variablen zusammen, die jeweils eine Maßnahmen-Asset-Kombination repräsentieren (z. B. *logPolicy1* implementiert auf *dbServerHosts*) und den Wert 1 annehmen, wenn die entsprechende Maßnahme auf das asset angewendet wird.

Zur Bewertung eines Maßnahmenportfolio wird das System zunächst entsprechend dem Genotyp initialisiert, anschließend werden eine Reihe von Angriffen mit unterschiedlichen Zufallsgenerator-Startwerten simuliert und die Ergebnisse aggregiert. Da wir Mehrfachzielsetzungen erlauben, wird in der Regel kein eindeutig “bestes” Maßnahmen-Portfolio gefunden, sondern eine Reihe von alternativen effizienten Lösungen. Jedes in dieser Menge enthaltene Portfolio ist nicht-dominiert, d.h. die Lösung enthält kein anderes Portfolio, das zumindest gleich gute Werte für alle Ziele und einen strikt besseren Wert für zumindest eines der Ziele aufweist.

Aufgrund des großen Entscheidungsraums und der “teuren” simulationsbasierten Evaluierungsprozedur handelt es sich um ein rechnerisch sehr aufwändiges Optimierungsproblem. Exakte Lösungen können daher nur für kleine Probleminstanzen durch vollständige Enumeration ermittelt werden. Für praxisrelevante Problemgrößen setzen wir daher genetische Algorithmen ein. Diese von natürlichen Evolutionsprozessen inspirierten Ansätze lösen Optimierungsprobleme implizit, indem sie eine Population von Individuen über mehrere

Generationen hinweg durch Bewertung ihrer „Fitness", Auswahl von „fitten" Individuen und Anwendung von Kreuzungs- und Mutationsoperatoren auf die ausgewählten Individuen entwickeln.

Konkret haben wir mit den Algorithmen NSGA-II [DPATM00] und SPEA-II [ZLT02] experimentiert und festgestellt, dass beide ein für unsere Zwecke zufriedenstellendes Leistungsverhalten zeigen. Für unser im nächsten Abschnitt vorgestelltes illustratives Beispiel haben wir Optimierungsläufe mit NSGA-II über 500 Generationen durchgeführt und typischerweise eine Konvergenz innerhalb der ersten 250 Generationen festgestellt.

5 Anwendungsbeispiel

5.1 Versuchsaufbau

Der Prototyp der Simulations- und Optimierungskomponenten wurden in Java implementiert. Die Simulation nutzt die Scheduling-Engine der Mason-Bibliothek [LCRPS04], die Wissensbasis wurde in SWI-Prolog [WSTL12] modelliert und der Zugriff darauf erfolgt über JPL [SDW]; die Optimierung mittels genetischer Algorithmen wurde mit Hilfe des Opt4j-Frameworks implementiert [LGRT11].

Für das Beispielszenario wurden 10 CAPEC-Angriffsmuster modelliert und um zusätzliche Muster ergänzt. Die im Anhang enthaltene Tabelle 3 fasst die modellierten Angriffsmuster zusammen. Außerdem wurden Abwehrmechanismen, wie etwa Antivirensoftware, Patches, Intrusion Detection-Systeme, Logging-Policies und Sicherheitstrainings, definiert. Die ebenfalls im Anhang enthaltene Tabelle 4 liefert eine Übersicht der modellierten Maßnahmen und deren Wirksamkeit in Bezug auf unterschiedliche Angriffe.

Schließlich wurde eine Organisation und deren IT-Infrastruktur durch Instanziierung von abstrakten Konzepten wie *host*, *subnet*, *data*, *user* etc. und deren Beziehungen (z.B. host *stores* data, host *uses* software) modelliert. Das Beispielszenario mit 30 Hosts, 5 Webservern, 5 Datenbankservern, 30 Angestellten und 3 Administratoren wurde automatisch von einem Generator erzeugt. Die im Anhang enthaltene Abbildung 4 stellt das modellierte System vereinfacht dar.

Wir simulieren fünf interne und externe Typen von Angreifern, die alle das Ziel verfolgen, Zugriff auf die Datenbank *db2* zu erlangen. Interne Angreifer verfügen bereits über (eingeschränkten) Zugang zum System; für externe Angreifer stellt die demilitarisierte Zone (DMZ) den zentralen Angriffspunkt dar. Ferner wird jeder Angreifer durch individuelle verhaltensbestimmende Attribute charakterisiert (siehe Tabellen 1, 2 und 3 im Anhang).

Für unsere Experimente wurden sechs Optimierungsziele definiert, nämlich Minimierung der Kosten, Minimierung erfolgreicher Angriffe, Maximierung der Entdeckung von Angriffen, und Minimierung des „Impacts" auf CIA-Attribute (Vertraulichkeit, Integrität, Verfügbarkeit) des Systems. Um die Beeinträchtigung der CIA-Attribute zu erfassen, wurden die Ausprägungen *low*, *medium* und *high* anhand einer lexikografischen Skala bemessen, d.h. lediglich die Anzahl der Ereignisse in der höchsten auftretenden Katego-

rie ist jeweils relevant. Insgesamt wurden 500 Generationen mit jeweils 30 Simulationsdurchgängen je Portfolio evaluiert.

5.2 Ergebnisse

Alle Berechnungen wurden auf einem 2x3GHz Xeon Prozessor mit einer Laufzeit zwischen 90 Minuten (*Admin*) und 50 Stunden (*Advanced Persistent Threat*) je Szenario durchgeführt. Der genetische Algorithmus identifizierte 251 effiziente Portfolios für das APT-Szenario, 306 für den versierten externen Angreifer, 104 für den unerfahrenen externen Angreifer, 58 für den Angestellten und 2 für den Administrator. Abbildung 2 zeigt die Zielwerte der effizienten Portfolios in einer Parallelkoordinaten-Darstellung. Die Y-Achse ist für die Impact-Attribute in die drei Kategorien *high* (oberhalb der roten Marke), *medium* (oberhalb der orangen Marke) und *low* (oberhalb der grünen Marke) untergliedert. Es wird jeweils nur die höchste Kategorie je Portfolio dargestellt.

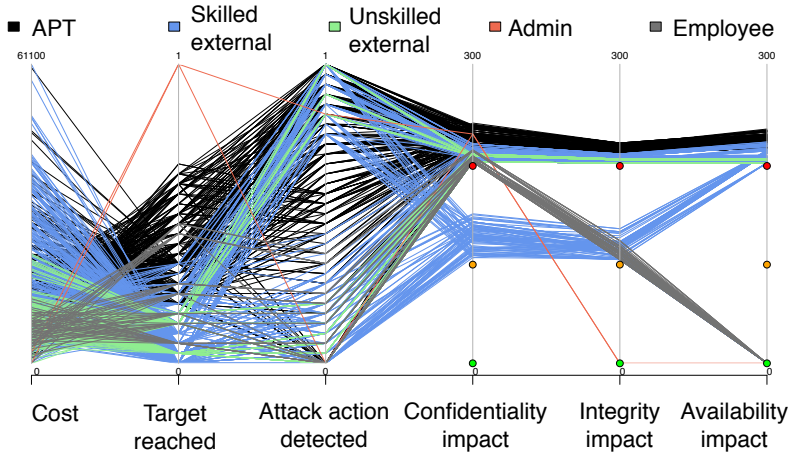


Abbildung 2: Zielwerte der effizienten Portfolios

Die im Anhang enthaltene Abbildung 5 gibt die Lösungsalternativen in einer Heatmap-Darstellung wieder. Die gefundenen effizienten Lösungen werden dabei zeilenweise dargestellt; der Genotypstring wird auf der linken Seite (blau) visualisiert und die Optimierungszielwerte für das Portfolio befinden sich rechts (rot). Jedes blaue Feld repräsentiert eine aktive Zuweisung einer Sicherheitsmaßnahme zu einem Asset. Die roten Ergebnisspalten für den Impact untergliedern sich in die Werte *low*, *medium* und *high*.

Die Ergebnisse dieses Beispiels zeigen, dass der Administrator als Angreifer das Ziel immer erreicht, da er bereits über die erforderlichen Rechte verfügt; eine Log-Policy auf *dbServerHost* erhöht aber zumindest die Entdeckungsrate. Ohne eingesetzte Sicherheitsmaßnahmen erreicht ein Angestellter das Ziel in etwa der Hälfte der Fälle. Die relativ hohe Erfolgsrate kann auf die Verfügbarkeit von effektiven sozialen Angriffsmöglichkeiten

zurückgeführt werden. Eine hohe Erfolgsrate von etwa zwei Drittel weist erwartungsgemäß der Angreifertyp *Advanced Persistent Threat* auf. Maßnahmenportfolios können den Anteil der erfolgreichen Attacken auf minimal 20% reduzieren bzw. die Entdeckungsrate bis auf 63% steigern. Der versierte externe Angreifer erreicht ähnliche Ergebnisse, wobei die Erfolgswahrscheinlichkeit etwas geringer ausfällt. Der durch einen unerfahrenen Angreifer verursachte Schaden ist weit weniger drastisch, und die Rate der erfolgreichen Angriffe kann mit geeigneten Sicherheitsmaßnahmen auf 3% reduziert werden. Gegen unerfahrene Angreifer zeigen Intrusion-Detection-Systeme eine sehr gute Wirkung. Bei erfahrenen Angreifern sind technische Kontrollen besonders oft in den Lösungsportfolios zu finden.

6 Zusammenfassung

Das vorgestellte Entscheidungsunterstützungssystem kann in komplexen Informationssystemen unter Bedachtnahme auf mehrfache Zielsetzung zur Erhöhung der IT-Sicherheit beitragen. Es greift auf eine Wissensbasis zurück, welche zu mehrstufigen Angriffen verkettbare Angriffsmuster modelliert und die Ermittlung möglicher Angriffspfade für unterschiedliche Angreifertypen ermöglicht. Darauf aufbauend kann eine Vielzahl von Angriffen für unterschiedliche Angreifertypen simuliert werden, um Systemkonfigurationen zu finden, die sich gegen diese Angreifer möglichst resistent zeigen. Die Ergebnisse unseres Experiments mit einer prototypischen Implementierung illustrieren den Nutzen für das IT-Sicherheitsmanagement.

Zukünftige Forschungsarbeit – aufbauend auf dem hier vorgestellten Ansatz – kann in mehrere Richtungen erfolgen. Die Entwicklung einer umfangreichen Sicherheitswissensbasis durch Erfassung weiterer Angriffsmuster ist ein naheliegender nächster Schritt. Diese Aufgabe ist keinesfalls trivial, erfordert sie doch die Entwicklung eines entsprechenden Vokabulars zur Spezifikation von Ursachen und Wirkungen über mehrere Abstraktionsebenen hinweg. Der Aufwand zur Abbildung formaler Angriffsmuster hängt von dem gewählten Detaillierungsgrad, sowie der Expertise des Modellierers ab. In diesem Beitrag wurde CAPEC zur Erstellung formaler Angriffsmuster herangezogen, es können aber natürlich auch andere Standards als Vorlage dienen. Entsprechend der Ausrichtung der Sicherheitsanalyse sollten technische, physische, und soziale Angriffsmuster einbezogen werden. Ferner wäre es interessant, unseren Ansatz in weiteren Szenarien mit strikten Sicherheitsanforderungen, wie beispielsweise im Bereich kritischer Infrastrukturen, zu testen. Schließlich könnte die Wissensbasis öffentlich zugänglich und von einer Community in einem gemeinsamen Repository weiterentwickelt werden. Organisationen müssten dann lediglich ihre eigene IT-Infrastruktur modellieren und könnten die gemeinsame Wissensbasis zur Optimierung ihres Systems nutzen. Da eine manuelle Modellierung und laufende Aktualisierung der IT-Infrastruktur sowohl fehleranfällig, als auch mit hohem Aufwand verbunden ist, sollten Ansätze und Werkzeuge zur automatisierten Inventarisierung betrachtet werden. Durch regelmäßige Updates könnten sie dieses außerdem laufend hinsichtlich der Widerstandsfähigkeit gegen neue Angriffstechniken testen, was den Informationsvorsprung zwischen Angreifer und Organisationen verringern würde.

ACKNOWLEDGMENTS

Die präsentierte Arbeit wurde im Rahmen des Forschungsprojektes "Moses3" vom Fonds zur Förderung der wissenschaftlichen Forschung (Austrian Science Fund FWF: P23122-N23) finanziert und bei Secure Business Austria, einem von der Österreichischen Forschungsförderungsgesellschaft FFG im Rahmen des COMET-Programm unterstützten K1 Kompetenzzentrums, umgesetzt.

Literatur

[AWK02] Paul Ammann, Duminda Wijesekera und Saket Kaushik. Scalable, graph-based network vulnerability analysis. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Seiten 217–224. ACM, 2002.

[BFP06] Stefano Bistarelli, Fabio Fioravanti und Pamela Peretti. Defense trees for economic evaluation of security investments. In *Proceedings of the First International Conference on Availability, Reliability and Security (ARES 2006)*, Seiten 416–423. IEEE Computer Society, 2006.

[Coh99] Fred Cohen. Simulating cyber attacks, defences, and consequences. *Computers & Security*, 18(6):479–518, 1999.

[CPJL01] Sung-Do Chi, Jong Sou Park, Ki-Chan Jung und Jang-Se Lee. Network security modeling and cyber attack simulation methodology. In *Proceedings of 6th Australasian Conference (ACISP 2001)*, 2001.

[DMCR06] G. C. Dalton, R. F. Mills, J. M. Colombi und R. A. Raines. Analyzing attack trees using generalized stochastic petri nets. In *IEEE Information Assurance Workshop*, Seiten 116–123, 2006.

[DPATM00] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal und T. T. Meyarivan. A fast elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2000.

[DW06] Ole Martin Dahl und Stephen D. Wolthusen. Modeling and execution of complex attack scenarios using interval timed colored Petri nets. In *Proceedings of the Fourth IEEE International Workshop on Information Assurance*, 2006.

[EFN09] Andreas Ekelhart, Stefan Fenz und Thomas Neubauer. AURUM - A framework for information security risk management. In *Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS)*, Seiten 1–10, Hawaii, USA, 2009.

[FE09] Stefan Fenz und Andreas Ekelhart. Formalizing Information Security Knowledge. In *Proceedings of the 4th ACM Symposium on Information, Computer, and Communications Security*, Seiten 183–194. ACM, 2009.

[KEG+14] Elmar Kiesling, Andreas Ekelhart, Bernhard Grill, Christine Strauss und Christian Stummer. Evolving secure information systems through attack simulation. In *Proceedings of the 2014 Hawaii International Conference on System Sciences (HICSS-47)*, Seiten 4868–4877. IEEE Computer Society, 2014.

[LCRPS04] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait und Keith Sullivan. MASON: A new multi-agent simulation toolkit. In *2004 SwarmFest Workshop*, 2004.

[LGRT11] Martin Lukasiewycz, Michael Glaß, Felix Reimann und Jürgen Teich. Opt4J: A modular framework for meta-heuristic optimization. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, Seiten 1723–1730. ACM, 2011.

[MN98] Makoto Matsumoto und Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, 1998.

[MRKS10] Carol Muehrcke, Elizabeth Ruitenbeek, Ken Keefe und William Sanders. Characterizing the behavior of cyber adversaries: The means, motive, and opportunity of cyber-attacks. In *40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Chicago, 2010.

[OBM06] Xinming Ou, Wayne F. Boyer und Miles A. McQueen. A scalable approach to attack graph generation. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, Seiten 336–345. ACM, 2006.

[Sch00] Bruce Schneier. *Secrets & Lies: Digital Security in a Networked World*. Wiley, New York, NY, 2000.

[SDW] Paul Singleton, Fred Dushin und Jan Wielemaker. JPL: A bidirectional Prolog/Java interface. Letzter Zugriff am 12. Februar 2014. `http://www.swi-prolog.org/packages/jpl/`.

[SHJ+02] Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann und Jeanette M. Wing. Automated generation and analysis of attack graphs. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Seiten 273– 284. IEEE, 2002.

[SO08] Reginald E. Sawilla und Xinming Ou. Identifying critical attack assets in dependency attack graphs. In *Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security*, ESORICS '08, Seiten 18–34. Springer, 2008.

[WSTL12] Jan Wielemaker, Tom Schrijvers, Markus Triska und Torbjörn Lager. SWI-Prolog. *Theory and Practice of Logic Programming*, 12(Special Issue 1-2):67–96, 2012.

[ZLT02] Eckart Zitzler, Marco Laumanns und Lothar Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. In K. Giannakoglou, D. Tsahalis, K. Papailiou und T. Fogarty, Hrsg., *Evolutionary Methods for Design, Optimisation and Control*. International Center for Numerical Methods in Engineering, 2002.

A Simulationsablauf

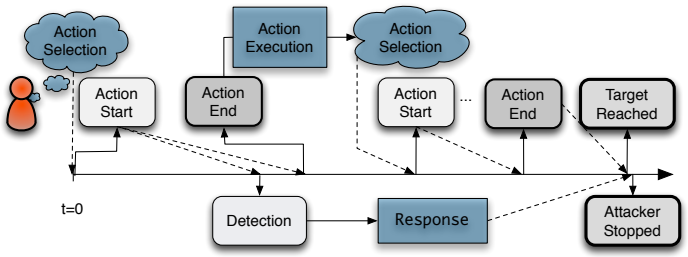


Abbildung 3: Ereignistypen und Simulationsablauf

B Angreiferprofile

Angreifertyp	Zeitlimit (Sek.)	w_{suc}	w_{det}	w_{dist}	Zugriff
Angestellter	150.000	0,45	0,25	0,30	interner Host
Administrator	300.000	0,50	0,20	0,30	alle Rechner
Versierter Externer	200.000	0,30	0,40	0,30	extern
Unerfahrener Externer	100.000	0,50	0,20	0,30	extern
Advanced Persistent Threat	1.000.000	0,50	0,20	0,30	extern

Einheitlich: $pContinueNew = 0,9$; $pRetry = 0,5$;
$pAlternativesNoNew = 0,7$; $pAlternativesFailed = 0,3$

Tabelle 1: Angreifertypen und Attribute

Angreifertyp	Verfügbare Angriffe
Angestellter (Skill-Level: 0)	shoulder surfing
Unerfahrener Externer (Skill-Level: 1)	spearfish attack
	sql injection
	social attack
	brute force
	email keylogger
	email backdoor
Versierter Externer (Skill-Level: 2)	(gleich wie Unerfahrener Externer)
	+ buffer overflow
	+ directory traversal
Administrator (Skill-Level: 2)	(gleich wie Versierter Externer)
Advanced Persistent Threat (Skill-Level: 3)	(gleich wie Versierter Externer)
	+ zero day

Tabelle 2: Verfügbare Angriffe

C Verhaltensmodell

Algorithm 1 Attacker behavior model

Input: available actions $\overline{A}$, previous action p, new actions $\overline{N}$
Output: selected action

```
1:  procedure SELECTACTION(Ā, p, N̄)
2:    for a ∈ Ā do
3:      if FULLFILLSTARGETCOND(ā) then
4:        return ā
5:      end if
6:    end for
7:    if p = null then
8:      CHOICE(Ā)
9:    else if WASSUCCESSFUL(p̄) then
10:     if |N| > 0 then
11:       if p_continueNew > RANDOM() then
12:         return CHOOSE(N̄)
13:       else
14:         return CHOOSE(Ā)
15:       end if
16:     else if p_altNoNew > RANDOM() then
17:       S̄ ← GETALTERNATIVES(p)
18:       return CHOOSE(S̄)
19:     else
20:       return CHOOSE(Ā)
21:     end if
22:   else if p_retry > RANDOM() then return p
23:   else if p_altFailed > RANDOM() then
24:     S̄ ← GETALTERNATIVES(p)
25:     return CHOOSE(S̄)
26:   else
27:     return CHOOSE(Ā)
28:   end if
29: end procedure
```

```
1: procedure CHOOSE(Ā)
2:   for ā ∈ Ā do
3:     d_ā^rel ← d(a,t) / (max(d(a,t))+1)
4:     W_ā ← p_success(ā)^w_success (1 − p_detection(ā))^w_detection (1 − d_ā^rel)^w_distance
5:   end for
6:   return WEIGHTEDCHOICE(Ā, W)
7: end procedure
```

D Angriffsmuster

Angriff	CAPEC (ID)
sql injection	SQL-Injektion (66)
social attack	Informationsgewinnung durch Social-Engineering (410)
brute force	Passwort Brute-Force Attacke (49)
spearfish attack	Informationsgewinnung durch gezielten persönlichen Angriff (407)
buffer overflow	Bufferüberlauf in API-Aufruf (8)
email backdoor	Email-Injektion (134)
zero day	Rechteausweitung (233)
directory traversal	Directory Traversal (213)
shoulder surfing	Informationsgewinnung über soziale Komponenten (404)
access data	legitimer Zugriff
access host	legitimer Zugriff

Tabelle 3: Modellierte Angriffsmuster

E Systemmodell

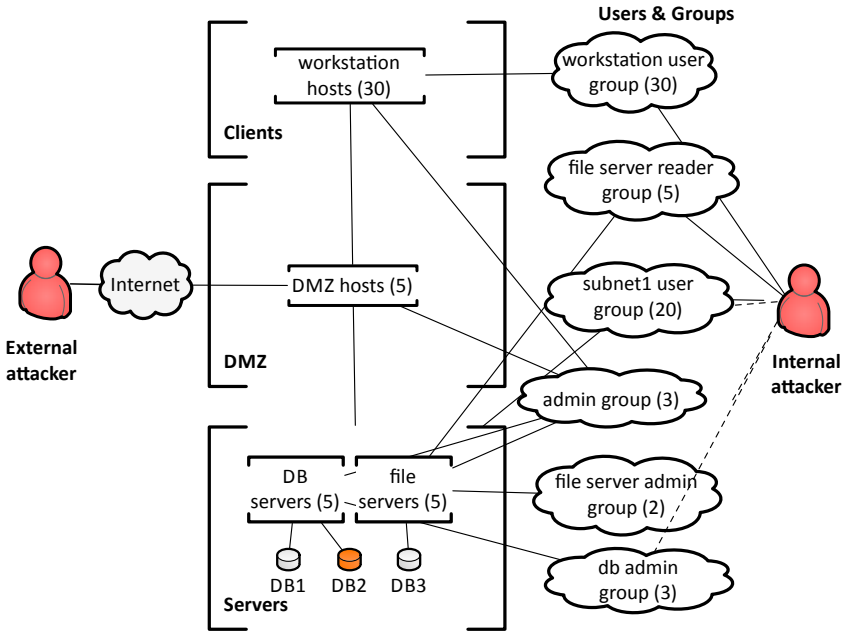


Abbildung 4: Systemmodell des Beispielszenarios

F Maßnahmen

Kategorie	Ausprägung	Angriff	Effektivität	Kosten
	AV1	buffer overflow	0.94	900
	AV1	email backdoor	0.70	900
	AV1	email keylogger	0.70	900
AV	AV1	zero day	0.05	900
	AV2	buffer overflow	0.98	1500
	AV2	email backdoor	0.85	1500
	AV2	email keylogger	0.85	1500
	AV2	zero day	0.08	1500
Code Review	Review1	sql injection	0.80	3200
Server Hardening	Hardening1	directory traversal	0.80	3000
	IDS1	buffer overflow	0.75	7500
	IDS1	sql injection	0.80	7500
IDS	IDS1	zero day	0.20	7500
	IDS2	buffer overflow	0.80	10000
	IDS2	sql injection	0.90	10000
	IDS2	zero day	0.25	10000
Log	Log	access data	0.80	2200
Patch	Patch	buffer overflow	0.98	400
	Train 1	email backdoor	0.30	1200
	Train 1	email keylogger	0.30	1200
	Train 1	shoulder surfing	0.30	1200
	Train 1	social attack	0.40	1200
	Train 1	spearfish attack	0.30	1200
	Train 2	email backdoor	0.70	1800
	Train 2	shoulder surfing	0.50	1800
Security Training	Train 2	email keylogger	0.70	1800
	Train 2	social attack	0.75	1800
	Train 2	spearfish attack	0.70	1800
	Train 3	email backdoor	0.80	4600
	Train 3	email keylogger	0.75	4600
	Train 3	shoulder surfing	0.75	4600
	Train 3	social attack	0.85	4600
	Train 3	spearfish attack	0.75	4600

Tabelle 4: Maßnahmen

G Simulationsergebnisse

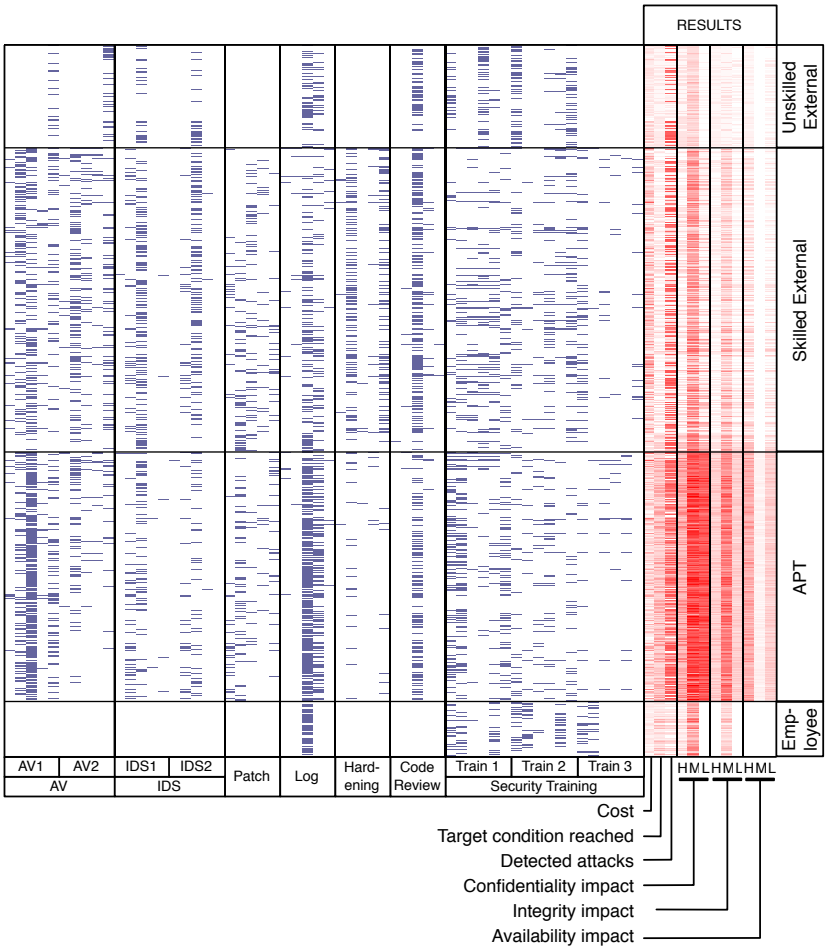


Abbildung 5: Eingesetzte Maßnahmen und Simulationsergebnisse

InnoDB Datenbank Forensik
Rekonstruktion von Abfragen über Datenbank-interne Logfiles

Peter Frühwirt[+], Peter Kieseberg[+], Christoph Hochreiner[+], Sebastian Schrittwieser[*], Edgar Weippl[+]

[+] SBA Research gGmbH
(pfruehwirt,pkieseberg,chochreiner,eweippl)@sba-research.org

[*] Fachhochschule St. Pölten GmbH
sebastian.schrittwieser@fhstp.ac.at

Abstract: Datenbanksysteme werden in der digitalen Forensik oft vernachlässigt, hinterlassen aber durch Transaktionen an vielen Stellen (temporäre) Spuren. Jede Transaktion kann - sofern unterstützt - rückgängig gemacht werden (rollback). InnoDB, eine populäre MySQL-Storage-Engine, erzeugt hierfür eigene Log-Dateien. In dieser Arbeit dekodieren wir diese internen Log-Files von InnoDB, um die ausgeführten Datenbankabfragen zu rekonstruieren, welche Daten verändert haben. Abschließend gehen wir noch auf die Zusammenhänge zwischen der internen Datenspeicherung und den Log-Dateien ein.

1 Einführung

Immer wenn Daten verarbeitet werden, gibt es viele Bereiche, wo Informationen temporär gespeichert werden [PS08]. Forensische Analysen können vergangene Aktivitäten zeigen, Zeitabläufe bzw. Teilzeitabläufe erstellen und gelöschte Daten wiederherstellen [SML10]. Während diese Tatsache in der Computer-Forensik bereits sehr lange bekannt ist und viele Ansätze [YYA09, FTB+06, MKY08, JL03] und Software Lösungen [FC05] existieren, befindet sich die systematische Analyse von Datenbanksystemen erst in den Kinderschuhen [WB08, Oli09]. InnoDB ist dabei eine der wenigen transaktionssicheren MySQL-Storage-Engines mit Unterstützung von commits, rollbacks und crash recovery [Corc, WA02].

Jede Änderung von Daten ist mit gleichzeitiger Ausführung einer Mini-Transaktion (mtr) implementiert [Tuu09b] und führt daher zu zumindest einem Aufruf der Funktion `mtr_commit()`, welche Daten in die InnoDB redo logs schreibt. InnoDB speichert keine Daten in einem bestimmten Datenformat in die Logs, sondern schreibt nur Abbilder bestimmter Speicherbereiche [Cora], wodurch sich die Komplexität der Rekonstruktion erhöht.

Erschwerend kommt hinzu, dass InnoDB die Logs in den sogenannten DoubleWrite Buffer [KYL+12] schreibt, welcher zu einem späteren Zeitpunkt die Daten gesammelt in das Log-File einträgt. Desweiteren benutzt InnoDB seit MySQL 5.1 einen optimierten Ansatz für die Kompression [mys08] bei der Speicherung.

2 Aufbau der Logs

Beim erste Start von MySQL erzeugt InnoDB zwei Logdateien *ib_logfile0* und *ib_logfile1* mit einer standardmäßigen Größe von je fünf Megabyte, sofern InnoDB mit der Option `innodb_file_per_table` konfiguriert wurde [Cord]. Beide Dateien besitzen dieselbe Struktur und werden von InnoDB abwechselnd beschrieben. Ähnlich der internen Datenspeicherung [FHMW10] sind die Log-Files in mehrere Fragemente unterteilt, welche in den folgenden Abschnitten beschrieben werden.

2.1 Datei-Header

Der erste Teil des Log-Files ist der Datei-Header, welcher allgemeine Informationen zur Datei beinhaltet. Er besitzt eine fixe Länge von 0x14 Bytes.

Tabelle 1 zeigt ein Beispiel für einen typischen Datei-Header.

Offset	Länge	Wert	Bedeutung
0x00	4	00 00 00 00	Gruppennummer der Log-Datei
0x04	8	00 00 00 00 01 F1 EA 00	Erste Log Sequence Number (LSN) der Log-Datei
0x0C	4	00 00 00 00	Nummer des archivierten Log-Files (0 wenn Log-Datei nicht archiviert wurde)
0x10	32	20 20 20 20 00 00 00 00 00 00 00 [...] 00 00 00 00 00 00 00 00	Diese Bytes werden von InnoDB-Hot-Backup benutzt. Sie beinhalten den Wert ”ibbackup” und den Zeitpunkt des erstellten Backups und werden für Informationen benutzt, welche dem Administrator angezeigt werden, wenn MySQL zum ersten Mal nach der Wiederherstellung gestartet wird.

Tabelle 1: Interpretation des Header-Blocks

2.2 Checkpoints und Crash recovery

InnoDB benutzt ein Checkpoint-System in den Log-Files und schreibt Modifikationen der Datenbank-Speicherseiten (*pages*) vom DoubleWrite Buffer in kleineren Schüben [KP05, Zai, BCKR02] auf die Festplatte, da eine Verarbeitung der Gesamtmenge die Ausführung

von SQL-Statements behindern würde.

InnoDB besitzt zwei Checkpoints in den Log-Files, welche abwechselnd beschrieben werden. Durch diese Methode wird garantiert, dass es immer mindestens einen vollständigen Checkpoint zur Wiederherstellung gibt. Während der Wiederherstellung nach einem Fehler [Tuu09a, Corb] lädt InnoDB die beiden Checkpoints und vergleicht deren ID-Labels. Jeder Checkpoint beinhaltet eine acht Byte lange *Log Sequence Number* (LSN). Diese LSN garantiert, dass alle Änderungen bis zu dieser Transaktionsnummer vorhanden sind. Dies bedeutet, dass alle Einträge kleiner als diese LSN auch im Log-File enthalten sind und kein Eintrag mit einer höheren ID existiert. Aus diesem Grund wird jede Änderung, welche nicht auf die Festplatte gespeichert wurde, in die crash-recovery-logs beziehungsweise in die Rollback-Logs gespeichert, InnoDB muss vor dem Schreiben auf die Festplatte einen Checkpoint erstellen [Tuu09a].
Die beiden Checkpoints befinden sich in den Log-Files *ib_logfile0* und *ib_logfile1* unter der Adresse 0x200 beziehungsweise 0x400. Jeder Checkpoint besitzt dieselbe Struktur und eine fixe Länge von 0x130 Bytes. Tabelle 3 zeigt eine detailierte Darstellung des Aufbaus eines Checkpoint-Blocks.

Die Daten des Checkpoints werden von der Methode `log_group_checkpoint()` [Tuu09b] (line 1675-1795) in den log group header geschrieben.

Offset	Länge	Wert	Bedeutung
0x00	8	00 00 00 00 00 00 00 12	Log checkpoint number
0x08	8	00 00 00 00 01 F3 64 92	Log sequence number des Checkpoints
0x10	4	00 00 0A 92	Offset zum log entry, berechnet durch `log_group_calc_lsn_offset()` [Tuu09b]
0x14	4	00 10 00 00	Größe des Buffers (fixer Wert: $2 \cdot 1024 \cdot 1024$)
0x18	8	FF FF FF FF FF FF FF FF	Archivierte log sequence number. Sofern `UNIV_LOG_ARCHIVE` nicht aktiviert ist, setzt InnoDB diese Felder auf FF FF FF FF FF FF FF FF.
0x20	288	00 00 [...] 00	Spacing und padding
0x120	4	28 E2 C3 AC	Prüfsumme 1 des gesamten Checkpoint-Blocks: 0x00-0x19F)
0x124	4	01 B0 72 29	Prüfsumme 2 über den Checkpoint-Block ohne LSN inklusive Prüfsumme 1: 0x08-0x124
0x128	4	00 00 00 05	Derzeitiges fsp (filespace) free limit im tablespace 0 (Einheit: MB). Diese Bytes werden nur von ibbackup benutzt, um zu entscheiden ob unbenutze Enden von non-auto-extending Datenfeldern abgeschnitten werden können [Tuu09c].
0x12C	4	55 E7 71 8B	Dieser Fixwert definiert, dass dieser Checkpoint die obigen Felder beinhaltet und wurde mit InnoDB-3.23.50 eingeführt [Tuu09c].

Tabelle 3: Interpretation des Checkpoints

2.3 Log Blöcke

Die Einträge in den Logdateien sind - im Gegensatz zu den Daten - nicht in pages sondern in Blöcken organisiert. Jeder Block besteht aus 512 Bytes Daten. Die Größe ist mit der Größe des Disk-Sektors identisch [Zai09]. Jeder Block besteht aus drei Teilen: Dem Log-Header, den eigentlichen Daten und dem Trailer. Diese Struktur dient der Erhöhung der Performance bei der Navigation in den Log-Files.

2.3.1 Log Block Header

Die ersten 12 Bytes eines Blocks werden *log block header* genannt. Dieser Block beinhaltet alle Informationen, die von InnoDB zur Verwaltung und zum Lesen der Log-Daten benötigt werden. Alle 1000 Bytes erstellt InnoDB automatisch einen neuen Header, daher gibt es in jedem Header einen Offset zum ersten Log-Eintrag innerhalb des nachfolgenden Datenblocks. InnoDB unterbricht bestehende Log-Einträge, um den Header an die dafür vorgesehene Stelle zu schreiben. Dies ermöglicht InnoDB schneller zwischen den Blöcken zu navigieren.

Offset	Länge	Wert	Bedeutung
0x00	4	80 00 F9 B2	Nummer des Log block headers. Wenn das Most Significant Bit (MSB) auf 1 gesetzt ist, ist der nachfolgende Block der erste Block im Log File. [Tuu09c]
0x04	2	00 EE	Anzahl an geschriebenen Bytes in diesem Block.
0x06	2	00 0C	Offset zum ersten Log-Eintrag in diesem Block.

Tabelle 4: Interpretation des log block headers

2.3.2 Log Block Trailer

Der Log Block Trailer (LBT) besteht aus einer vier Byte langen Prüfsumme zur Verifikation der Gültigkeit des Blocks.

Offset	Länge	Wert	Bedeutung
0x00	4	36 9E 2E 96	Prüfsumme des Log Block Inhaltes. In älteren InnoDB Versionen (vor Version 3.23.52) wurde an dieser Stelle derselbe Wert wie `LOG_BLOCK_HDR_NO` anstelle der Prüfsumme gespeichert [Tuu09c].

Tabelle 5: Interpretation des log block trailers

3 Log-Einträge

Die nachfolgenden Demo-Rekonstruktionen der Datenbankabfragen basieren auf dem folgenden Datenmodell (Listing 1).

Listing 1: Verwendete Tabellenstruktur

```
CREATE TABLE `fruit3` (
  `primaryKey` int(10) NOT NULL,
  `field1` varchar(255) NOT NULL,
  `field2` varchar(255) NOT NULL,
  `field3` varchar(255) NOT NULL,
  PRIMARY KEY (`primaryKey`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Wir haben zwei einfache Datentypen (Integer und Varchar) benutzt, um die Vorgehensweise bei der Rekonstruktion zu demonstrieren. InnoDB speichert einen Integer-Wert in einem Feld mit einer fixen Länge von 4 Bytes, das Datenformat Varchar ist hingegen von variabler Länge. Diese beiden Datentypen decken damit die Gestalt der Logfiles für nahezu alle anderen Typen ab und unterscheiden sich von diesen lediglich in der Länge des verwendeten Speicherplatzes und der Interpretation der Daten. Für unsere forensische Analyse ist das Wissen über die Tabellenstruktur erforderlich. Dieses Wissen kann aus den Informationen der `.frm` Dateien [FHMW10] bzw. den Speicherdumps aus den Log-Files gewonnen werden.

4 Rekonstruktion von SQL-Queries

In diesem Abschnitt werden SQL-Statements mit Hilfe der InnoDB-Log-Files rekonstruiert. Zu beachten ist, dass einige Daten in komprimierter Form gespeichert sind. Die Längenangaben dieser Werte sind in den Tabellen mit Sternen versehen und können daher abhängig vom konkreten SQL-Statement teilweise erheblich abweichen. Ältere Versionen von InnoDB schrieben die Logs ohne Komprimierung und verbrauchten dadurch wesentlich mehr Speicher. Auf die genauere Definition und Analyse dieses Dateiformats für forensische Zwecke wird in dieser Arbeit nicht eingegangen.

Für jede SQL-Abfrage, welche Daten ändert, erzeugt InnoDB zumindest einen `MLOG_UNDO_INSERT` Log-Eintrag. Dieser Eintrag speichert die betroffene Table Identification Number (TIN), den Typ der Abfrage (INSERT, UPDATE, DELETE) sowie typspezifische Informationen.

4.1 Insert

InnoDB erstellt für eine einzelne INSERT-Abfrage insgesamt 9 Log-Einträge. Diese sind alle mit `MLOG_` als Prefix versehen.

1. `8BYTES`
2. `UNDO_HDR_REUSE`
3. `2BYTES`
4. `2BYTES`
5. `2BYTES`
6. `MULTI_REC_END`
7. `UNDO_INSERT`
8. `COMP_REC_INSERT`
9. `2BYTES`

Die meisten dieser Log-Einträge sind dabei vom Typ `MLOG_2BYTES` beziehungsweise `MLOG_8BYTES` und zeigen lediglich kleinere Änderungen in Dateien an. Für die forensische Analyse sind nur die Einträge `MLOG_UNDO_INSERT` und `MLOG_COMP_REC_INSERT` interessant.

In diesem Abschnitt beschreiben wir die letzten Bestandteile des Log-Eintrags `MLOG_COMP_REC_INSERT`. Im Vorhinein wurde bereits die referenzierte Tabelle im `MLOG_UNDO_INSERT`-Eintrag ausgelesen. Die Tabellenstruktur ist daher bereits vor dem Auslesen der eigentlichen Logdaten, über die Referenz zur manipulierten Tabelle, bekannt.

Offset	Länge	Wert	Bedeutung
0x00	1	26	Log-Typ des Eintrags (`COMP_REC_INSERT`)
0x01	1*	58	Tablespace-ID
0x02	1*	03	Page-ID
0x03	2	00 06	Anzahl der Felder in diesem Eintrag s (n)
0x05	2	00 01	Anzahl der einzigartigen Felder (n_unique)
0x07	2	80 04	Länge des ersten Datenfeldes (primaryKey).
0x09	2	80 06	Länge des zweiten Datenfeldes (transactionID)
0x0B	2	80 07	Länge des dritten Datenfeldes (data rollback pointer)
0x0D	2	80 00	Länge des vierten Datenfeldes (Zu beachten ist, dass bei diesem Datenfeld keine Längenangabe existiert, da es sich um ein Feld von dynamischer Länge handelt (Varchar). Das hat zur Folge, dass es im späteren Verlauf noch eine Größenangabe für dieses Feld geben muss)
0x0F	2	80 00	Länge des fünften Datenfelds (field2)
0x11	2	80 00	Länge des sechsten Datenfelds (field3)
0x13	2	00 63	Offset
0x15	1*	59	Länge des letzten Segments (*end segments*)
0x16	1	00	Info- und Status-Bits
0x17	1*	08	Offset zum Ursprung
0x18	1	00	Mismatch index
0x19	1*	04	Länge des ersten Felds mit variabler Länge (field1)
0x1A	1*	05	Länge des zweiten Felds mit variabler Länge (field2)
0x1B	1*	0A	Länge des dritten Felds mit variabler Länge (field3)
0x1C	5	00 00 30 FF 56	Unbekannt
0x21	4	80 00 00 04	PrimaryKey
0x25	6	00 00 00 00 40 01	Transaction-ID
0x2B	7	00 00 00 00 33 21 28	Data rollback pointer

0x31	10*	73 74 72 61 77 62 65 72 72 79	Daten des ersten Felds (”strawberry”)
0x3B	5*	61 70 70 62 65	Daten des zweiten Felds (”apple”)
0x40	4*	6B 69 77 69	Daten des dritten Felds (”kiwi”)

Die Struktur des `MLOG_COMP_REC_INSERT` Eintrags ist sehr kompliziert. Nach dem allgemeinen Datenblock (logType, spaceID und pageID), welcher die verwendete Tabelle beinhaltet, finden sich die folgenden Längeninformationen: n und n_unique. Der Wert n gibt an, aus wie vielen Feldern dieser Log-Eintrag besteht, n_unique definiert die Anzahl der Primärschlüssel. Anschließend an diese beiden Feldern folgen jeweils n-mal 2 Bytes Längeninformationen zu den jeweiligen Datenfeldern, wobei die ersten für die Primärschlüssel reserviert sind (n_unique Felder).

Sollte die Feldlänge den Wert 0x8000 besitzen, bedeutet dies, dass die Feldlänge dynamisch berechnet wird und vom Datensatz abhängig ist (zum Beispiel im Fall eines Feldes vom Typ VARCHAR). Für alle statischen Datentypen (z.B. INTEGER) definiert dieser Wert die Länge des Feldes. Zusätzlich muss beachtet werden, dass in diesen Längendefinitionen auch systeminterne Felder, wie die Transaktions-ID und der data rollback pointer definiert werden.

In den nächsten sieben* Feldern befinden sich Metadaten, die für die Rekonstruktion nicht relevant sind. Darauf anschließend findet sich wieder Informationen, welche für die Berechnung der Länge des Datensatzes entscheidend sind. Die tatsächliche Länge wird hier für jeden Eintrag mit dynamischer Länge spezifiziert.

Die folgenden fünf Bytes werden für diverse flags benötigt. In den ersten drei Blöcken befinden sich die eigentlichen Log-Daten, beginnend mit dem Primärschlüssel. Die Anzahl der Felder des Primärschlüssels s wird dabei am Anfang im Feld n_unique definiert. Durch die Typinformationen aus der .frm-Datei [FHMW10] ist bekannt, dass es sich hierbei um einen Primärschlüssel bestehend aus einer INTEGER-Spalte handelt. Anhand dieser Information kann die Länge und der enthaltene Wert berechnet werden.

Die nächsten beiden zwei Felder sind systeminterne Felder: Die Transaktions-ID und der Data Rollback Pointer (DRP). Der DRP referenziert die Position in der Datei, an welche der Datensatz geschrieben wurde. Wenn zum Beispiel ein Dateneintrag mit Hilfe einer UPDATE-Operation verändert wird, besitzt dieser Log-Eintrag denselben data rollback pointer wie der Ursprungseintrag. Die letzten Felder beinhalten die eigentlichen Daten der jeweiligen Spalten, welche nicht Teil des Primärschlüssels sind. Die Anzahl der Felder ist immer (n_unique - 2), da die beiden internen Felder (DRP und Transaktions-ID) bereits behandelt wurden. In dem angeführten Beispiel besitzen die Felder eine dynamisch allokierte Länge (Typ: VARCHAR). Aus diesem Grund wird die Länge nicht ausschließlich in den Feldern über die Längenangaben definiert (Einträge nach dem Feld n_unique). Es gibt das zuvor erwähnte Feld für die dynamischen Längen. Mit Hilfe dieser Informationen können die Daten der Felder rekonstruiert werden: ”strawberry”, ”apple”, ”kiwi”.

Wir kennen daher den Primärschlüssel sowie die Werte der Datenfelder, in Kombination mit den gesammelten Informationen über die Tabellenstruktur aus der .frm-Datei kann die ausgeführte Datenbankabfrage wiederherstellt werden (Listing 2).

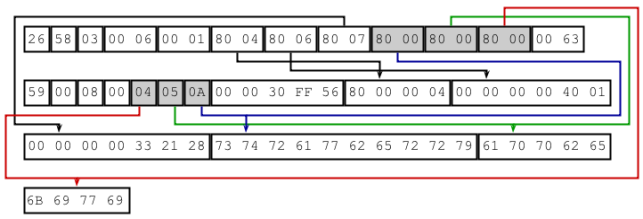


Abbildung 1: Zusammenhänge zwischen den Datenfeldern in einem MLOG_COMP_REC_INSERT Log-Eintrag

Listing 2: Wiederhergestellte INSERT-Abfrage

```
INSERT INTO forensic1.fruit3 (primaryKey, field1, field2, field3)
       VALUES (4, 'strawberry','apple','kiwi');
```

4.2 Update

Die Rekonstruktion einer UPDATE-Operation verläuft ähnlich wie die eines INSERT-Statements. Der einzige Unterschied ist, dass sich zusätzliche, für die forensische Analyse interessante Informationen extrahieren lassen. In unseren Untersuchungen haben wir nur UPDATE-Operationen betrachtet, die den Primärschlüssel nicht verändern, da diese keine Änderungen am Index vornehmen und dadurch weitere Änderungen in den Log-Dateien verursachen.

4.2.1 Wiederherstellung von überschriebenen Daten

InnoDB speichert die überschriebenen Daten für die Wiederherstellung nach einem Ausfall der Datenbank oder zur Durchführung von Rollbacks. Zunächst muss für die Analyse dieser Daten der `MLOG_UNDO_INSERT` Eintrag betrachtet werden.

Offset	Länge	Wert	Bedeutung
0x00	1	94	Typ des Eintrags (UNDO_INSERT)
0x01	1*	00	Tablespace-ID
0x02	1*	33	Page-ID
0x03	2	00 1B	Länge des Log-Eintrags
0x05	1	1C	Typ der Datenmanipulation (Update eines bestehenden Eintrags)
0x06	2	00 68	ID der Tabelle
0x08	6	00 00 00 00 40 01	Letzte Transaktionsnummer des veränderten Feldes
0x0D	5*	00 00 33 21 28 04	Bisheriger DRP
0x13	1	04	Länge des Primärschlüssels
0x14	4	80 00 00 04	Betroffener Primärschlüssel

0x18	1	01	Anzahl der geänderten Felder
0x19	1	04	FieldID
0x1A	1	05	Länge des Feldes
0x1B	5	61 70 70 6C 65	Überschriebener Wert des Feldes

Die ersten drei Bytes sind wie in jedem Log-Eintrag gleich aufgebaut: Typ, SpaceID und PageID, die Bytes vier und fünf definieren die Länge des Log-Eintrags, daher ist das Ende des Eintrags sehr einfach zu bestimmen. Die folgenden Bytes definieren den Typ des *undo-insert*-Typs. Diese Klassifikation ist notwendig, da der *undo-insert*-Typ nur besagt, dass irgendetwas in einem Eintrag geändert wurde. Die folgenden Daten sind abhängig von diesem Typ und müssen gesondert interpretiert werden. In dem vorhandenen Beispiel betrachten wir nun den Typ 12 (`TRX_UNDO_UPDATE_EXIST_RECORD`) näher. Der Typ wird berechnet, indem der Wert modulo 16 genommen wird.

Die ersten beiden Bytes bestimmen die betroffene Tabelle (table identifier). Dieses Identifizierungsmerkmal findet man auch in den Tabellendefinitionen der `.frm`-Dateien an der Adresse 0x26. In Kombination mit diesen Informationen erhält man den Namen der Tabelle. In dem vorhandenen Beispiel handelt es sich um die Tabelle ”fruit3”. Die folgenden sechs beziehungsweise sieben Bytes beinhalten die Transaktions-ID und den DRP der Datenfelder. Die Transaktions-ID referenziert auf die letzte ausgeführte Transaktion vor diesem Update. In diesem Beispiel handelt es sich um die INSERT-Operation, die diesen Eintrag angelegt hat. Mit Hilfe dieser Transaktionsnummer kann die komplette Historie eines Datensatzes, beginnend bei seiner Erstellung, rekonstruiert werden.

In den nachfolgenden Bytes wird der Primärschlüssel definiert. Abhängig von der Anzahl der Schlüssel wird dieser Schritt mehrfach wiederholt. Daher ist es wichtig vorab zu wissen, aus wie vielen Feldern der Primärschlüssel der Tabelle besteht. Das erste Byte definiert die Länge des Primärschlüsselfeldes. In diesem Beispiel handelt es sich um einen Integer, daher beinhaltet dieses Feld den Wert vier. Die nächsten *n* Bytes, wobei n von der vorherigen Längenangabe bestimmt wird, speichern den konkreten Wert des Primärschlüsselfeldes: vier ($0x8004$ *signed*).
Die nächsten Bytes bestimmen die Anzahl der Spalten m (in dem vorhandenen Beispiel ist $m = 1$), die durch die UPDATE-Operation verändert wurden.

Aus diesem Grund muss der nachfolgende Schritt *m*-mal wiederholt werden. Dabei besitzt jeder Datenblock dieselbe Struktur: Das erste Byte speichert die Feldnummer, das zweite Feld die Länge des Datenblocks und die nachfolgenden Bytes die eigentlichen Daten. In unserem Beispiel wurde der Wert von Feld Nr. 4 (”field2”) mit dem vorherigen Wert 0x61 70 70 6C 65 (”apple”) überschrieben.

4.3 Delete

Die Wiederherstellung von DELETE-Operationen verläuft ähnlich wie bei UPDATE Operationen. Es muss zwischen DELETE-Operationen, welche einen Datensatz wirklich physikalisch löschen und solchen, die den Datensatz nur als gelöscht markieren unterschieden werden.

Ein Log-Eintrag, der einen Eintrag lediglich als gelöscht markiert, ist sehr kurz und besteht aus 4 Teilen. Für die forensische Rekonstruktion ist hier jedoch nur ein einziger Block interessant: `MLOG_UNDO_INSERT`. Die folgende Tabelle zeigt diesen Log-Eintrag für eine durchgeführte DELETE-Operation.

Offset	Länge	Wert	Bedeutung
0x00	1	94	Typ des Log-Eintrags (UNDO_INSERT)
0x01	1*	00	Tablespace-ID
0x02	1*	33	Page-ID
0x03	2	00 1E	Länge des Log-Eintrags
0x05	1	0E	Typ der Datenmanipulation (delete record)
0x06	2	00 66	Table-ID
0x08	6	00 00 00 00 28 01	Letzte Transaktionsnummer des gelöschten Feldes
0x0D	8*	E0 80 00 00 00 2D 01 01 10	Bisheriger DRP
0x17	1	04	Länge des Primärschlüssels
0x18	4	80 00 00 04	Betroffener Primärschlüssel
0x1B	3	00 08 00	Unknown
0x1E	1	04	Länge des Primärschlüsselfeldes
0x20	4	80 00 00 01	Primärschlüssel des gelöschten Eintrages

Analog zu den UPDATE-Operationen muss die Anzahl der Primärschlüssel bereits bekannt sein. Dieser Log-Eintrag speichert im Prinzip nur den Primärschlüssel des gelöschten Datensatzes und kann über das Dateisystem wiederhergestellt werden [FHMW10]. Der komplette Datensatz wird nur dann im Log gespeichert, wenn der Datensatz physisch gelöscht wird. Die ausgeführte Abfrage kann analog zu den INSERT- bzw. UPDATE-Operationen wiederhergestellt werden.

Listing 3: Rekonstruierte DELETE-Query

```
DELETE FROM forensic1.fruit3 WHERE primaryKey=1;
```

5 Fazit

InnoDB verwendet interne Log-Dateien, um Transaktionen zurückzusetzen (Rollback) beziehungsweise zur Wiederherstellung nach einem Crash. Die gespeicherten Log-Einträge beinhalten Kopien von Teilen des Speicherbereichs sowie von Daten. In dieser Arbeit wurde gezeigt, wie SQL-Operationen aus den internen Log-Dateien der Datenbank wiederhergestellt werden können. Die demonstrierten Techniken können genutzt werden, um eine komplette Transaktionsgeschichte zu rekonstruieren. Wir haben eine bekannte Struktur der Tabellen vorausgesetzt, welche wir durch unsere bisherigen Arbeiten [FHMW10, FKS+13] bereits forensisch gewinnen konnten. Zusätzlich wurden Operationen, die die Tabellenstruktur verändern (z.B. ALTER TABLE), vernachlässigt. Durch die gewonnenen Informationen aus den Log-Dateien lassen sich diese Informationen jedoch analog wiederherstellen.

Literatur

[BCKR02] Ryan Bannon, Alvin Chin, Faryaaz Kassam und Andrew Roszko. InnoDB Concrete Architecture. *University of Waterloo*, 2002.

[Cora] Oracle Corporation. Compression and the InnoDB Log Files (accessed 31.01.2014). `http://dev.mysql.com/doc/innodb-plugin/1.0/en/innodb-compression-internals-log.html`.

[Corb] Oracle Corporation. InnoDB Checkpoints (accessed 31.01.2014). `http://dev.mysql.com/doc/mysql-backup-excerpt/5.1/en/innodb-checkpoints.html`.

[Corc] Oracle Corporation. Storage Engines (accessed 31.01.2014). `http://dev.mysql.com/doc/refman/5.1/en/storage-engines.html`.

[Cord] Oracle Corporation. Using Per-Table Tablespaces (accessed 31.01.2014). `http://dev.mysql.com/doc/refman/5.5/en/innodb-multiple-tablespaces.html`.

[FC05] Guillermo Francia und Keion Clinton. Computer forensics laboratory and tools. *Journal of Computing Sciences in Colleges*, 20(6), June 2005.

[FHMW10] Peter Frühwirt, Markus Huber, Martin Mulazzani und Edgar Weippl. InnoDB Database Forensics. In *Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010)*, 2010.

[FKS+12] Peter Frühwirt, Peter Kieseberg, Sebastian Schrittwieser, Markus Huber und Edgar Weippl. InnoDB Database Forensics: Reconstructing Data Manipulation Queries from Redo Logs. In *5th International Workshop on Digital Forensic*, 2012.

[FKS+13] Peter Frühwirt, Peter Kieseberg, Sebastian Schrittwieser, Markus Huber und Edgar Weippl. InnoDB database forensics: Enhanced reconstruction of data manipulation queries from redo logs. *Information Security Technical Report*, 2013.

[FTB+06] Guillermo Francia, Monica Trifas, Dorothy Brown, Rahjima Francia und Chrissy Scott. Visualization and management of digital forensics data. In *Proceedings of the 3rd annual conference on Information security curriculum development*, 2006.

[JL03] Hongxia Jin und J. Lotspiech. Forensic analysis for tamper resistant software. *14th International Symposium on Software Reliability Engineering*, November 2003.

[KP05] Michael Kruckenberg und Jay Pipes. *Pro MySQL*. Apress, 2005.

[KYL+12] Woon-Hak Kang, Gi-Tae Yun, Sang-Phil Lim, Dong-In Shin, Yang-Hun Park, Sang-Won Lee und Bongki Moon. InnoDB DoubleWrite Buffer as Read Cache using SSDs. In *10th USENIX Conference on File and Storage Technologies*, 2012.

[MKY08] J. Todd McDonald, Yong Kim und Alec Yasinsac. Software issues in digital forensics. *ACM SIGOPS Operating Systems Review*, 42(3), April 2008.

[mys08] Changes in Release 5.1.x (Production). http://dev.mysql.com/doc/refman/5.1/en/news-5-1-x.html, 2008.

[Oli09] Martin Olivier. On metadata context in Database Forensics. *Digital Investigation*, 4(3-4), March 2009.

[PS08] Kyriacos Pavlou und Richard Snodgrass. Forensic analysis of database tampering. *ACM Transactions on Database Systems (TODS)*, 33(4), November 2008.

[SML10] Patrick Stahlberg, Gerome Miklau und Brian Neil Levin. Threats to privacy in the forensic analysis of database systems. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2010.

[Tuu09a] Heikki Tuuri. Crash Recovery and Media Recovery in InnoDB. In *MySQL Conference*, April 2009.

[Tuu09b] Heikki Tuuri. MySQL Source Code (5.1.32). /src/storage/innobase/log/log0log.c, 2009.

[Tuu09c] Heikki Tuuri. MySQL Source Code (5.1.32). /src/storage/innobase/include/log0log.h, 2009.

[WA02] Michael Widenius und David Axmark. *MySQL reference manual: documentation from the source*. O'Reilly, 2002.

[WB08] Paul Wright und Donald Burleson. *Oracle Forensics: Oracle Security Best Practices (Oracle In-Focus series)*. Paperback, 2008.

[YYA09] Pei-Hua Yen, Chung-Huang Yang und Tae-Nam Ahn. Design and implementation of a live-analysis digital forensic system. In *Proceedings of the 2009 International Conference on Hybrid Information Technology*, 2009.

[Zai] Peter Zaitsev. MySQL Performance Blog - InnoDB Double Write (accessed 31.01.2014). `http://www.mysqlperformanceblog.com/2006/08/04/innodb-double-write/`.

[Zai09] Peter Zaitsev. Innodb Architecture and Performance Optimization. In *O'Reilly MySQLConference and Expo*, April 2009.

Fingerprinting Techniques for Target-oriented Investigations in Network Forensics

Dominik Herrmann, Karl-Peter Fuchs, and Hannes Federrath
Vogt-Kölln-Straße 30, D-22527 Hamburg
{herrmann, fuchs, federrath}@informatik.uni-hamburg.de

Abstract: Fingerprinting techniques are receiving widespread attention in the field of information security. In this paper we argue that they may be of specific interest for the field of network forensics. In three case studies, we explore the use of fingerprinting techniques to improve and extend current investigative methods and showcase why fingerprinting allows for more target-oriented investigations than current practices. In each case study, we review the applicability of the current state of the art from the field of information security. The paper is intended to be a starting point for a discussion about the opportunities and concerns that may result from using evidence gained by fingerprinting techniques in criminal investigations.

1 Introduction

In the late 19th century it was discovered that fingerprints of humans are a distinctive biometric trait [Fau80, Gal92]. Today fingerprints play an important role in criminal investigations, and they serve as convincing evidence for the association of a suspect with a crime in court all over the world.

Apart from the biometric fingerprints mentioned above the computer science community is also aware of *device fingerprints* that capture characteristic traits of devices in a technical system. The notion of such fingerprints came up in the era of the Cold War, where defense forces became interested in deducing the type and make of missiles and satellites solely based on their radio echo. It was found that different devices exhibit distinctive patterns, "radar fingerprints", that can be observed during their flight [Lac67].

Fingerprinting techniques are increasingly gaining attention: According to Google Scholar, in the years 2010, 2011 and 2012 more than 200 security-related academic papers containing the term "fingerprinting" in the title have been published annually. The motives for studying these techniques relate to both, their opportunities, e. g., their utility for network intrusion detection systems [HBK04] or as an additional authentication factor [XGMT09], as well as to the risks entailed, e. g., data leakage [DCGS09] or infringement of personal privacy [BFGI11].

Based on talks with lawyers and investigators we believe that the utility of fingerprinting for criminal investigations is relatively unknown so far. Historically, computer forensics was mainly concerned with analyzing hard disks found in computers seized by means of a warrant, e. g., after raiding the home of a purported suspect [Cas09]. However, due to full-

disk encryption (FDE), obtaining incriminating data from seized hard disks has become more difficult recently [CFGS11]. Moreover, data is increasingly stored "in the cloud", where legal accessibility often depends on coordination of and cooperation between authorities and service providers in multiple countries. Due to unresolved legal issues relating to such multi-jurisdiction investigations policy makers are concerned that investigators may be barred from obtaining relevant pieces of evidence in the future [CL10].

Due to this development investigators have started to turn their attention to the field of *network forensics*, which studies the probative value of network traffic in criminal investigations [Cas04, DH12]. For this purpose law enforcement agencies (LEAs) have several investigative measures at their disposal (in the following, we use Germany as an example): Firstly, given an IP address that has been involved in criminal activities they can *request customer subscription data from an ISP* to determine the identity of the respective account holder ("Bestandsdatenauskunft", cf. § 113 TKG and § 100j StPO). Secondly, they can request an ISP to perform *lawful interception*, i. e., to store and disclose the network traffic of a specific customer ("Telekommunikationsüberwachung", cf. §§ 100 a and 100 b StPO). Having access to the network traffic, investigators are facing **two common challenges**: Firstly, they may want to establish an association between criminal activities detected on the network and an actual perpetrator **(C1: "who did it?")**. Secondly, they may be struggling to find evidence for criminal activities, when traffic is encrypted during transport **(C2: "what was done?")**.

Further, policy makers around the world have tried to overcome these challenges by introducing new data retention laws ("Vorratsdatenspeicherung") as well as legalizing the use of police-operated malware to intercept traffic before it is encrypted and sent over the network ("Bundestrojaner"). Critics argue that such efforts are disproportionate, because they lay the ground for uncontrolled surveillance, which also resulted in strict decisions of the German constitutional court (BVerfG, 27.2.2008 – 1 BvR 370/07; BVerfG, 2.3.2010 – 1 BvR 256/08). Before considering the deployment of disproportionate measures we suggest to fully leverage the potential of already existing investigative measures that are both, more target-oriented and incident-related. In this paper we demonstrate in three case studies that the application of fingerprinting techniques can play a vital role in this regard.

The rest of the paper is organized as follows: We outline the commonly used fingerprinting approach in Sect. 2, proceeding with our three case studies in Sects. 3, 4 and 5. After that, we discuss limitations and concerns in Sect. 6, before we conclude the paper in Sect. 7.

2 Fundamentals: Fingerprinting and Related Techniques

Fingerprinting techniques allow for the (re-)identification of a subject or an object (or "entity" in general) based on characteristic traits, i. e., its *fingerprint*. In contrast to *explicit* identifiers such as a serial number the fingerprint typically captures *implicit properties*.

Typically the fingerprinting approach consists of two stages. In the first stage, also called *training stage*, the fingerprints of a set of entities are recorded and stored together with the identity of the entity in a database. In the second stage, the *identification stage*, the

characteristics of an entity whose identity is unknown are observed. The identity of the unknown entity is then inferred by comparing its fingerprint with all known fingerprints.

Fingerprinting research has been called both, an *art* as well as *engineering* [TDH03]. The art refers to designing a set of *suitable features* that facilitates identification of entities. Like in biometrics, good features are *distinctive* and *permanent* at the same time [JRP04], i. e., the features of each entity are characteristic enough so that one can differentiate various entities based on them, while one given entity exhibits the same (or at least a very similar) fingerprint every time it is encountered. On the other hand, the efficient extraction and robust matching of fingerprints is facilitated by the application of concepts drawn from engineering, e. g., pattern matching, statistical modeling and machine learning.

Watermarking and Correlation Techniques We point out that the fingerprinting approach outlined above is different from *watermarking techniques* [WCJ07, HB11, HKB12, HB13], which allow an investigator to trace back received packets to their true source, even when an adversary obfuscates his identity by using multi-hop anonymization networks such as Tor (http://torproject.org, [DMS04]) or compromised hosts, so-called "stepping stones" [YE00, ZP00]. To this end the investigator *tags* singular network flows at one location, e. g., by means of artificially delaying packets, and tries to detect them again at a different location, e. g., on the dial-up line of a household. If the watermark is embedded at a location, where the fact that all network traffic relates to criminal activities is known for sure (e. g., on the uplink of an online shop for hard drugs), the investigator can use this information as indication that a suspect has been involved in criminal activities.

Watermarking techniques require active interference in network communications, which may be problematic from a legal perspective. Although less effective, *flow correlation techniques* [WRW02, JWJ+13] may be more suitable for the purpose of establishing an association between criminal activities and a suspect, because they can be applied by a *passive* observer. Flow correlation techniques extract and match *already existing* patterns by observing packet timings and traffic volumes. However, like watermarking techniques they require the investigator to have the capability to observe traffic at (at least) *two locations* in the network at the same time: Apart from intercepting the traffic of the suspect – which is also a requirement for fingerprinting techniques – investigators have to additionally tap the traffic of the server, whose location is typically either unknown (e. g., in the "cloud", where virtual machines are relocated on the fly) or under the authority of a foreign jurisdiction. Therefore, the utility of watermarking and flow correlation techniques for criminal investigations is more limited in comparison to fingerprinting techniques, which can be applied by a passive observer with access to a single site only.

3 Case Study 1: Inferring Content of Encrypted Communications

In this first case study we demonstrate the utility of fingerprinting for Challenge C2: "what was done?". We assume a scenario in which investigators suspect that Mallory, whose identity has been disclosed to police by his ISP, is consuming child pornography on the

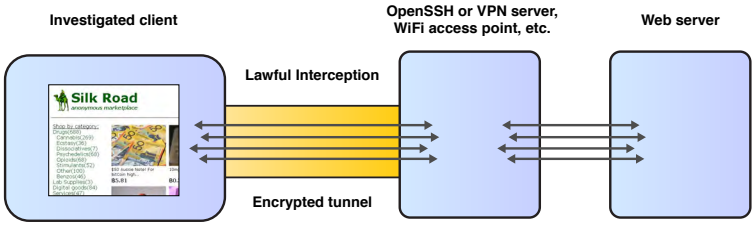


Figure 1: Website Fingerprinting Scenario

Internet. The prosecutor wants to obtain corroborating evidence that confirms the accusations before bringing the case to court. Thus, the objective of the prosecution is to find evidence that specific activities have been carried out by the suspect, e. g., that he visited a website, downloaded a movie or entered specific search terms into a search engine [OLL11]. A classical investigative measure involving digital forensics would consist in raiding Mallory's apartment, seizing his computer and trying to extract the contents of the browser history and cache files [Per09]. However, in our scenario no evidence can be obtained, because Mallory uses FDE on his machine.

As an alternative measure investigators might opt for lawful interception, i. e., compel Mallory's ISP to tap and disclose his network traffic to the LEA. However, in our case study this approach will not result in any corroborative evidence, either, because Mallory makes sure that all potentially incriminating activity is perpetrated via encrypted channels: He uses the VPN service of an offshore provider whenever he connects to the Internet, and "serious business" is conducted only via the Tor network (cf. Fig. 1).

Apparently, in such a scenario the only opportunity for law enforcement to obtain evidence for Mallory's criminal involvement seems to consist in questionable measures like infecting his machine with police-operated malware to perform a remote search. However, there are fingerprinting techniques that can be applied by investigators to infer (part of) Mallory's activities, e. g., whether he accessed a specific site or content, solely based on the traffic metadata that is not encrypted. This approach is known as *traffic analysis* [Ray00].

3.1 Packet-based Website Fingerprinting

Packet-based website fingerprinting exploits the fact that many websites are a unique composition, consisting of multiple source and media files. The concrete number and size of the individual files of a site has been shown to be a distinctive and permanent feature suitable for fingerprinting [Hin02, SSW+02]. When a site is downloaded within an encrypted channel such as a VPN, the sizes of individual files cannot be determined by an observer anymore. However, fingerprints can still be built from features derived from the encrypted TCP flows. *Inter-arrival times* of packets have been shown to be characteristic [BLJL05], but due to network latency this feature is not very robust. The *distribution of the size of the IP packets* is a more promising candidate (cf. Fig. 2).

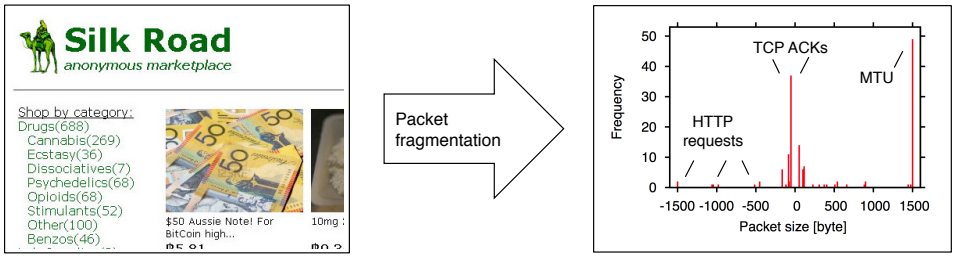


Figure 2: Characteristic distribution of IP packet sizes due to packet fragmentation [HWF09]

In [LL06] Liberatore and Levine, who focus on IP packet sizes and packet direction, have shown for a sample of 1000 sites that observers can infer the websites downloaded via OpenSSH tunnels with a Naïve Bayes classifier or the Jaccard index. Their best technique scores a website identification accuracy of 73 %. In a similar experiment inspired by their work we have improved the detection accuracy to 97 % by applying suitable transformations to the raw data [HWF09]. Our results indicate that fingerprinting is not limited to OpenSSH tunnels, but also effective for other popular techniques (e. g., IPsec and OpenVPN). Based on our dataset, the work of Panchenko et al. is one of the first to demonstrate that fingerprinting may also work in the Tor network [PNZE11]. Since then this problem (also termed "encrypted page identification" [XRYX13]) has received much attention (cf. [DCRS12] and [WG13] for an overview).

3.2 DNS-based Website Fingerprinting

DNS-based website fingerprinting serves a different purpose. It can be useful for forensic investigators that need to determine the **search terms** of a suspect whose traffic has been seized using lawful interception. Until recently, traffic to web search engines was not encrypted. However, in 2013 the leading search provider Google enabled SSL for all queries by default, which obscures the entered query terms in the intercepted traffic. The only remaining option for investigators to obtain the queries of a suspect consists in forcing Google to disclose them, which may be infeasible due to legal cross-border issues.

However, under certain circumstances DNS-based website fingerprinting can be used to infer the search terms entered into search engines, even when requests to the search engine are encrypted with HTTPS. This possibility was first studied by Krishnan and Monrose [KM10, KM11], who present results for the Firefox and Chrome browsers with the Google search engine. DNS-based fingerprinting exploits the fact that some browsers not only issue the DNS queries required to load the content embedded in the currently rendered site, but they also issue DNS queries for the links (`<a>` tags) that are contained in a site. The rationale for this *DNS pre-resolution* is to speed up the download of the next site when the user clicks a link. In addition, the Chrome browser tries to guess the links the user will most likely click on, in order to download the corresponsing sites in the background (DNS queries due to "site prefetching").

Table 1: Inferring Search Terms via DNS-based Website Fingerprinting on Google (HTTPS enabled)

Browser	Link pre-resolution	Site prefetching	Speculative resolution
Firefox 25.0.1	no (HTTP only)	no	yes
Chrome 31.0.1650.57	no (HTTP only)	yes	yes
Safari 6.1 (8537.71)	no (HTTP only)	no	no

Krishnan and Monrose demonstrate that DNS-based fingerprinting works especially well for search engines, whose primary purpose is to display lists of links for the relevant sites and for ads. For many search queries the set of sites displayed on the first results page is a distinctive and stable feature. In order to infer the search terms by DNS-based fingerprinting, the investigator would have to issue the interesting queries himself and establish a database containing the observed DNS queries. Inferring the search terms may also be feasible if the database is incomplete, because in some cases the DNS queries issued by the browser *literally repeat* the search terms (e. g., searching for "gun powder" may induce the search engine to insert an ad for the site *gunpowder.buycheaper.biz*).

Moreover, some browsers, i. e., Firefox, Chrome, and Safari, allow the user to enter search terms directly into the address bar. This comfortable feature can also cause the search terms to be leaked via DNS queries, when the browser cannot rule out that an entered keyword *is* the hostname of a web server (*speculative resolution*).

The effectiveness of DNS-based website fingerprinting is subject to technical **limitations**. First of all, search engines do not necessarily display the same search results to all users, i. e., it may be difficult to match search queries based on the observed DNS queries. Moreover, in recent versions browser manufacturers have turned off pre-resolution for HTTPS sites for privacy reasons, i. e., the pre-resolution based technique is not effective any more. However, in our own experiments (Nov. 2013) we could still observe DNS queries due to speculative resolution and due to site prefetching (cf. Table 1).

4 Case Study 2: Device-based Ascription of Activities

In the second case study we showcase how fingerprinting techniques can be leveraged to address Challenge C1: "who did it?". We assume that law enforcement has managed to gain authority over an underground marketplace that specializes in illegal drugs. Instead of taking the site down, investigators intercept and retain all incoming traffic on the web servers for a while in order to obtain new leads for tracking down the dealers. To this end they request customer subscription data from the respective ISPs for the observed source IP addresses. Of particular interest is one of the more active customers, Carol, who shares a broadband Internet connection with her unsuspecting flat mate Alice. Based on the intercepted traffic, investigators *do* know that the criminal activity originated from their apartment, however they *do not* know who of the two flat mates is involved. Police forces raid the apartment and seize the laptops of Alice and Carol, who deny any involvement in criminal activities. The forensic analysis of the hard disks does not produce any evidence,

Table 2: Device fingerprinting with individual (I) and class (C) characteristics

	Trait	Involvement	Source
I	Skew of real-time clock [KBC05]	active probing	Manuf.
C	TCP stack of operating system [CL94]	active probing	OS
C	TCP stack of operating system [Bev04]	passive logging	OS
C	JS benchmark [MBYS11], HTML 5 rendering [MS12]	active website	HW/SW
I	Browser fingerprints [Eck10]	active website	SW
C	TCP flow characteristics during surfing [YHMR09]	passive logging	SW

either: Alice, whose laptop is secured using FDE, denies to disclose her password, and on Carol's machine no traces of activity can be found, because she always entered the "private browsing" mode before engaging in unlawful endeavors.

However, in this scenario investigators may still be able to determine the computer from which the criminal activity originated, because different devices may exhibit different behavior on the network, resulting in a characteristic *device fingerprint*. Device fingerprints are present due to tolerances in the *manufacturing process* and differences in *hardware and software implementations*. They can either capture **class characteristics**, which are emitted by all devices that share the same specification, or **individual characteristics**, which allow to uniquely identify a single device [Cas11, 17].

In order to leverage device fingerprints to ascribe the criminal activities to one of the two computers in question, investigators have to extract the browser and operating system fingerprints from the incoming requests while they intercept the traffic of the underground marketplace. After the raid investigators can compare the fingerprints they observed for the requests that led them to Alice and Carol (e. g., "Firefox browser on a Windows PC") with the hard- and software configuration of the two suspects. If only one of the two computers is found to match the specification indicated by the fingerprint, this can serve as evidence for involvement.

In the following we will briefly survey the most relevant device fingerprinting approaches before we provide results from our own experiments with DNS-based fingerprinting.

4.1 Existing Device Fingerprinting Techniques

Table 2 showcases the landscape of fingerprinting techniques that could be leveraged by investigators that want to associate a specific device with certain actions. **Active device fingerprinting** techniques provoke a device to emit its fingerprint in response to specially crafted probing packets or by explicitly reading out certain properties. Comer and Lin were among the first to observe that the TCP stacks of operating systems respond differently when they receive spurious packets and that they also vary in terms of timeouts and retransmission behavior [CL94], which can serve as a class characteristic. Kohno et al. discovered that multiple desktop machines can be remotely differentiated due to skew in their real-time clocks [KBC05]. While this constitutes a powerful individual characteristic, it cannot be observed passively in a reliable manner. The trend for tighter integration

of browsers with the operating system has also created new fingerprinting opportunities that emit class characteristics: Mowery et al. show that hardware/software configurations can be either differentiated with a crafted JavaScript benchmark embedded in a website [MBYS11] or by analyzing the anti-aliasing artifacts of text that is rendered using the *canvas* tag in HTML 5 [MS12]. Summarizing the results of the "EFF Panopticlick" experiment Eckersly illustrates that individual browsers – and sometimes even devices – can be re-identified with high probability by a website that actively enumerates the list of installed browser plug-ins and system fonts [Eck10].

A limitation of active techniques is the fact that they require *active involvement* on the part of the investigator. In contrast, **passive device fingerprinting** needs only traits that are "voluntarily" provided by a device and thus collected easily. A well-known property of this kind is the "user agent" string in HTTP requests, which explicitly states browser and operating system. The information provided there can be forged easily, though, i. e., its probative value is questionable. More reliable are unavoidably exhibited implicit behavioral traits, e. g., the class characteristics that allow for passive operating system detection with the tool *p0f* (http://lcamtuf.coredump.cx/p0f3/) based on characteristic implementations of the TCP/IP stack [Bev04]. Similar techniques can be applied to differentiate various browser types [YHMR09].

4.2 DNS-based Device Fingerprinting

DNS-based device fingerprinting is a passive technique that exploits the fact that operating systems and browsers can be identified via the hostnames they resolve during regular background activities, e. g., time synchronization and polling for software updates [MYK13]. The application of this technique requires LEAs to access the DNS queries of a suspect, either by way of lawful interception or by legally forcing the ISP to record and disclose the DNS traffic of a suspect. Investigators can cross-correlate the class characteristics inferred by DNS-based fingerprinting with other evidence. For instance, they can detect faked user agent strings, if there are no DNS queries that conform to the stipulated operating system or browser. The results of our experiments, in which we observed the background traffic of various browsers on multiple platforms, indicate that DNS-based device fingerprinting is feasible for common configurations (cf. Tables 3 and 4 in the Appendix).

5 Case Study 3: Behavior-based Ascription of Activities

In the third case study we assume a similar scenario as in Case Study 2, however, this time investigators do not succeed in extracting device fingerprints from the requests issued to the underground marketplace. Nevertheless, they may still be able to solve Challenge C1 ("who did it?") by applying *behavior-based fingerprinting* techniques, which allow to link multiple surfing sessions of the same user as well as to distinguish sessions of different users solely based on (alleged) characteristic web usage patterns.

In this case we assume that investigators have access to Alice's and Carol's traffic via lawful interception. Starting off with an anonymous behavioral fingerprint of a surfing session that contains an incriminating action (the "initial fingerprint"), the objective of the investigator consists in identifying additional surfing sessions of the (still not identifiable) suspect that match the initial fingerprint ("suspect's sessions"). Note that it is not required that all sessions linked to the initial fingerprint contain instances of incriminating behavior; they will be useful for investigators even if they contain only innocuous behavior.

Once a sufficiently large number of suspect's sessions has been acquired, there are two conditions that may indicate to investigators that Carol, and not Alice, is the offender: either an *intersection attack* or *unintentional identity disclosure*. For an intersection attack the investigators analyze the temporal usage patterns that emerge from the suspect's sessions in order to infer at what times the offender was online. This information can be used in concert with classical investigative techniques, such as physical observation or questioning, to infer the identity of the offender, e. g., because Alice may have an alibi for some of the points in time. Intersection attacks have been applied with success in various contexts [BL02, KAP02, KAPR06, PGT12]. The second condition occurs when Carol discloses her identity in one of her sessions that have been linked to the suspect's sessions via the initial fingerprint, e. g., by logging on to a shopping portal or mail account with her personal credentials. If the credentials are unavailable, e. g., due to encryption with HTTPS, investigators can still compel the respective service provider to disclose the identity of the account based on the observed date and time of the log-in.

Note that investigators will only be able to extract the "initial fingerprint" if Carol engages at least one more time in incriminating activity (e. g., by visiting the underground marketplace) after lawful interception of traffic has commenced. In some cases, such as our drug trafficking scenario, this is a reasonable assumption.

Previous research suggests that behavior-based ascription of activities is possible in practice, i. e., multiple sessions of the same user can be linked due to characteristic website visitation patterns. The **behavioral fingerprint** of a user consists of the set of visited websites, which are the result of the distinct set of his personal interests. The potential of behavioral fingerprints for differentiating users can already be appreciated by cursory visual inspection of individual sessions (see Fig. 3).

Yang shows how machine learning techniques used for *association rule mining* can be leveraged to extract and match such behavioral patterns [Yan10]. In a controlled setting with up to 100 concurrent users the predictive accuracy of her techniques reaches up to 87 % when profiles are built using 100 training sessions per user. Accuracy drops to only 62 % if only a single training session is available.

However, according to our own research, which has been published in detail in [BHF12, HBF13], extracting and matching behavioral fingerprints *is* feasible even when only a single session is available. We reach this conclusion based on our experiments with an anonymized dataset that contains the DNS queries issued by a set of 3862 enrolled students over the course of two months. For each user we extract a behavioral fingerprint (queried DNS hostnames and access frequencies) from one of his sessions, which is then used to identify the remaining sessions of the same user. With a Multinomial Naïve Bayes

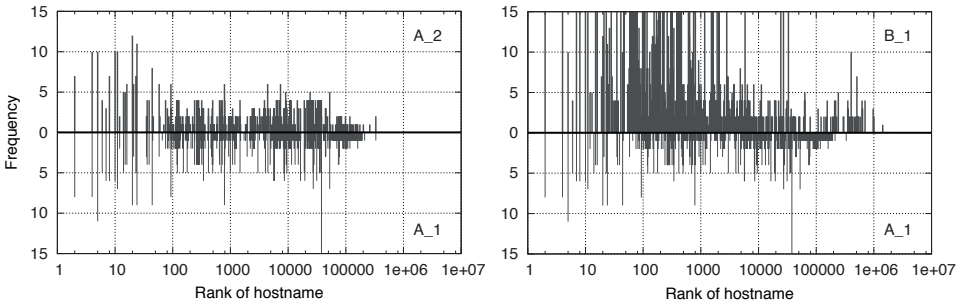


Figure 3: Higher visual similarity between website visitation behavior in two sessions of the same user (left-hand side) in comparison to sessions of different users (right-hand side)

classifier [MRS08, 258] we obtain cross-validated precision and recall values beyond 70 % for sessions of up to 24 hours and 3000 concurrent users. Further experiments have shown that accuracy of linking improves for smaller sets of users, e. g., it reaches 90 % for 100 concurrent users.

6 Limitations and Concerns

While we argue that fingerprinting techniques may be a suitable tool to improve the forensic tool-chain we stress that they are also subject to considerable limitations and thus have to be adopted with care in criminal investigations.

Firstly, the probative value of the results obtained by fingerprinting is often unclear. On the one hand, techniques that involve machine learning algorithms may suffer from poor explainability, when the investigator cannot fully explain how the classifier reached a decision. On the other hand, high accuracy values obtained in scientific papers may convey a certain level of confidence and objectivity, but usually they are only meant to give an indication of the performance of a technique for a specific closed-world scenario in a controlled setting, i. e., it is unknown how they perform "in the wild". Standardized corpora and strict evaluation methodologies have to be established to increase the confidence of the results.

Secondly, once law enforcement agencies have learned to appreciate the qualities of fingerprinting techniques for the purpose of criminal investigations, security policy makers may move forward and call for mandatory online dragnet investigations based on fingerprints, which could even be argued to be "privacy-preserving" – because only abstract fingerprints and not the actual content are screened, after all. Nevertheless such measures would be the equivalent of pre-emptive blanket surveillance. Therefore, instead of only focusing on the improvement of fingerprinting techniques researchers should also work on usable countermeasures to keep the balance.

7 Conclusions

In this paper, we have explored the adoption of state of the art fingerprinting techniques from information security research to the field of network forensics. Our analysis suggests that fingerprinting can be used to establish associations between criminal activities and their perpetrators as well as to find corroborative evidence for criminal activities even if only encrypted traffic can be observed. In two of our three case studies (cf. Sects. 3 and 5) we have shown that evidence can be obtained with fingerprinting that would otherwise require measures of questionable proportionality, such as data retention or remote search via state malware. In our third scenario (cf. Sect. 4) no client-side lawful interception is required at all.

These findings suggest that, instead of deploying heavy artillery, the adoption of fingerprinting techniques may lead the way to more target-oriented and incident-related investigations in the future and might also become a viable enhancement for current investigative methods. However, several challenges must be met before fingerprinting will become dependable and practical for these scenarios.

Acknowledgements

This paper was inspired by talks and discussions at Dagstuhl Seminar 13482 on Forensic Computing. The authors are grateful to all participants for their insightful ideas and fruitful feedback. We also thank our colleague Christoph Gerber for inspiring our discussions of the case studies.

References

[Bev04] R. Beverly. A Robust Classifier for Passive TCP/IP Fingerprinting. In C. Barakat and I. Pratt, editors, *PAM*, volume 3015 of *LNCS*, pages 158–167. Springer, 2004.

[BFGI11] K. Boda, A. M. Földes, G. G. Gulyás, and S. Imre. User Tracking on the Web via Cross-Browser Fingerprinting. In P. Laud, editor, *NordSec*, volume 7161 of *LNCS*, pages 31–46. Springer, 2011.

[BHF12] C. Banse, D. Herrmann, and H. Federrath. Tracking Users on the Internet with Behavioral Patterns: Evaluation of Its Practical Feasibility. In D. Gritzalis, S. Furnell, and M. Theoharidou, editors, *SEC*, volume 376 of *IFIP Advances in Information and Communication Technology*, pages 235–248. Springer, 2012.

[BL02] O. Berthold and H. Langos. Dummy Traffic Against Long Term Intersection Attacks. In Dingledine and Syverson [DS03], pages 110–128.

[BLJL05] G. D. Bissias, M. Liberatore, D. J. Jensen, and B. N. Levine. Privacy Vulnerabilities in Encrypted HTTP Streams. In G. Danezis and D. Martin, editors, *Privacy Enhancing Technologies*, volume 3856 of *LNCS*, pages 1–11. Springer, 2005.

[Cas04] E. Casey. Network traffic as a source of evidence: tool strengths, weaknesses, and future needs. *Digital Investigation*, 1(1):28–43, 2004.

[Cas09] E. Casey. *Handbook of Digital Forensics and Investigation*. Academic Press, 2009.

[Cas11] E. Casey. *Digital Evidence and Computer Crime – Forensic Science, Computers and the Internet*. Academic Press, 3rd edition, 2011.

[CFGS11] E. Casey, G. Fellows, M. Geiger, and G. Stellatos. The growing impact of full disk encryption on digital forensics. *Digital Investigation*, 8(2):129–134, 2011.

[CL94] D. Comer and J. C. Lin. Probing TCP Implementations. In *USENIX Summer*, pages 245–255, 1994.

[CL10] D. D. Clark and S. Landau. The Problem Isn't Attribution: It's Multi-stage Attacks. In *Proceedings of the Re-Architecting the Internet Workshop*, ReARCH '10, pages 11:1–11:6, New York, NY, USA, 2010. ACM.

[DCGS09] M. Dusi, M. Crotti, F. Gringoli, and L. Salgarelli. Tunnel Hunter: Detecting Application-Layer Tunnels with Statistical Fingerprinting. *Computer Networks*, 53(1):81–97, 2009.

[DCRS12] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *IEEE Symposium on Security and Privacy*, pages 332–346. IEEE, 2012.

[DH12] S. Davidoff and J. Ham. *Network Forensics: Tracking Hackers through Cyberspace*. Pearson Education, 2012.

[DMS04] R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium*, pages 303–320. USENIX, 2004.

[DS03] R. Dingledine and P. F. Syverson, editors. *Privacy Enhancing Technologies, Second International Workshop, PET 2002, San Francisco, CA, USA, April 14-15, 2002, Revised Papers*, volume 2482 of *LNCS*. Springer, 2003.

[Eck10] P. Eckersley. How Unique Is Your Web Browser? In M. J. Atallah and N. J. Hopper, editors, *Privacy Enhancing Technologies*, volume 6205 of *LNCS*, pages 1–18. Springer, 2010.

[Fau80] H. Faulds. On the Skin-Furrows of the Hand. *Nature*, 22(574):605, 1880.

[Gal92] F. Galton. *Finger Prints*. Macmillan and Co., London, 1892.

[HB11] A. Houmansadr and N. Borisov. SWIRL: A Scalable Watermark to Detect Correlated Network Flows. In *NDSS*. The Internet Society, 2011.

[HB13] A. Houmansadr and N. Borisov. The Need for Flow Fingerprints to Link Correlated Network Flows. In E. De Cristofaro and M. Wright, editors, *Privacy Enhancing Technologies*, volume 7981 of *LNCS*, pages 205–224. Springer, 2013.

[HBF13] D. Herrmann, C. Banse, and H. Federrath. Behavior-based tracking: Exploiting characteristic patterns in DNS traffic. *Computers & Security*, 39A:17–33, November 2013.

[HBK04] J. Hall, M. Barbeau, and E. Kranakis. Enhancing Intrusion Detection in Wireless Networks Using Radio Frequency Fingerprinting. In M. H. Hamza, editor, *Communications, Internet, and Information Technology*, pages 201–206. IASTED/ACTA Press, 2004.

[Hin02] A. Hintz. Fingerprinting Websites Using Traffic Analysis. In Dingledine and Syverson [DS03], pages 171–178.

[HKB12] A. Houmansadr, N. Kiyavash, and N. Borisov. Non-blind Watermarking of Network Flows. *CoRR*, abs/1203.2273, 2012.

[HWF09] D. Herrmann, R. Wendolsky, and H. Federrath. Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial Naïve-Bayes Classifier. In R. Sion and D. Song, editors, *CCSW*, pages 31–42. ACM, 2009.

[JRP04] A. K. Jain, A. Ross, and S. Prabhakar. An Introduction to Biometric Recognition. *IEEE Trans. Circuits Syst. Video Techn.*, 14(1):4–20, 2004.

[JWJ+13] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. F. Syverson. Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries. In A. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM Conference on Computer and Communications Security*, pages 337–348. ACM, 2013.

[KAP02] D. Kesdogan, D. Agrawal, and S. Penz. Limits of Anonymity in Open Environments. In F. A. P. Petitcolas, editor, *Information Hiding*, volume 2578 of *LNCS*, pages 53–69. Springer, 2002.

[KAPR06] D. Kesdogan, D. Agrawal, D. V. Pham, and D. Rautenbach. Fundamental Limits on the Anonymity Provided by the MIX Technique. In *IEEE Symposium on Security and Privacy*, pages 86–99. IEEE, 2006.

[KBC05] T. Kohno, A. Broido, and K. C. Claffy. Remote Physical Device Fingerprinting. In *IEEE Symposium on Security and Privacy*, pages 211–225. IEEE, 2005.

[KM10] S. Krishnan and F. Monrose. DNS Prefetching and Its Privacy Implications: When Good Things Go Bad. In *Proceedings of the 3rd USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More*, LEET'10, Berkeley, CA, USA, 2010. USENIX Association.

[KM11] S. Krishnan and F. Monrose. An Empirical Study of the Performance, Security and Privacy Implications of Domain Name Prefetching. In *DSN*, pages 61–72. IEEE, 2011.

[Lac67] E. A. Lacy. Radar Signature Analysis. *Electronics World*, 77(2):23–26, February 1967.

[LL06] M. Liberatore and B. N. Levine. Inferring the Source of Encrypted HTTP Connections. In A. Juels, R. N. Wright, and S. De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 255–263. ACM, 2006.

[MBYS11] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham. Fingerprinting Information in JavaScript Implementations. In *Proceedings of Web 2.0 Security and Privacy 2011 (W2SP)*, San Franciso, May 2011.

[MRS08] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008.

[MS12] K. Mowery and H. Shacham. Pixel Perfect: Fingerprinting Canvas in HTML5. In M. Fredrikson, editor, *Proceedings of W2SP 2012*. IEEE, May 2012.

[MYK13] T. Matsunaka, A. Yamada, and A. Kubota. Passive OS Fingerprinting by DNS Traffic Analysis. In L. Barolli, F. Xhafa, M. Takizawa, T. Enokido, and H. Hsu, editors, *AINA*, pages 243–250. IEEE, 2013.

[OLL11] J. Oh, S. Lee, and S. Lee. Advanced Evidence Collection and Analysis of Web Browser Activity. *Digital Investigation*, 8:62–70, August 2011.

[Per09] M. T. Pereira. Forensic Analysis of the Firefox 3 Internet History and Recovery of Deleted SQLite Records. *Digital Investigation*, 5(3-4):93–103, 2009.

[PGT12] F. Pérez-González and C. Troncoso. Understanding Statistical Disclosure: A Least Squares Approach. In S. Fischer-Hübner and M. Wright, editors, *Privacy Enhancing Technologies*, volume 7384 of *LNCS*, pages 38–57. Springer, 2012.

[PNZE11] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel. Website Fingerprinting in Onion Routing Based Anonymization Networks. In Y. Chen and J. Vaidya, editors, *WPES*, pages 103–114. ACM, 2011.

[Ray00] J. Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In H. Federrath, editor, *Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 10–29. Springer, 2000.

[SSW+02] Q. Sun, D. R. Simon, Y. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu. Statistical Identification of Encrypted Web Browsing Traffic. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Washington, DC, USA, 2002. IEEE.

[TDH03] K. I. Talbot, Paul R. Duley, and M. H. Hyatt. Specific Emitter Identification and Verification. *Technology Review Journal*, 2003.

[WCJ07] X. Wang, S. Chen, and S. Jajodia. Network Flow Watermarking Attack on Low-Latency Anonymous Communication Systems. In *IEEE Symposium on Security and Privacy*, pages 116–130. IEEE, 2007.

[WG13] T. Wang and I. Goldberg. Improved Website Fingerprinting on Tor. In A. Sadeghi and S. Foresti, editors, *WPES*, pages 201–212. ACM, 2013.

[WRW02] X. Wang, D. S. Reeves, and S. F. Wu. Inter-Packet Delay Based Correlation for Tracing Encrypted Connections through Stepping Stones. In D. Gollmann, G. Karjoth, and M. Waidner, editors, *ESORICS*, volume 2502 of *LNCS*, pages 244–263. Springer, 2002.

[XGMT09] L. Xiao, L. J. Greenstein, N. B. Mandayam, and W. Trappe. Fingerprints in the Ether: Using the Physical Layer for Wireless Authentication. *CoRR*, abs/0907.4877, 2009.

[XRYX13] W. Xia, Y. Ren, Z. Yuan, and Y. Xue. TCPI: A Novel Method of Encrypted Page Identification. In *1st International Workshop on Cloud Computing and Information Security (CCIS 2013)*. Atlantis Press, 2013.

[Yan10] Y. Yang. Web user behavioral profiling for user identification. *Decision Support Systems*, 49:261–271, 2010.

[YE00] K. Yoda and H. Etoh. Finding a Connection Chain for Tracing Intruders. In F. Cuppens, Y. Deswarte, D. Gollmann, and M. Waidner, editors, *ESORICS*, volume 1895 of *LNCS*, pages 191–205. Springer, 2000.

[YHMR09] T. Yen, X. Huang, F. Monrose, and M. K. Reiter. Browser Fingerprinting from Coarse Traffic Summaries: Techniques and Implications. In U. Flegel and D. Bruschi, editors, *DIMVA*, volume 5587 of *LNCS*, pages 157–175. Springer, 2009.

[ZP00] Y. Zhang and V. Paxson. Detecting Stepping Stones. In *Proceedings of the 9th Conference on USENIX Security Symposium*, pages 13–13, Berkeley, CA, USA, 2000. USENIX Association.

A Results for DNS-based Device Fingerprinting

In order to evaluate the feasibility of DNS-based device fingerprinting we recorded the hostnames that are resolved autonomously in the background by commonly used web browsers and desktop operating systems. To this end we have performed experiments with MS Internet Explorer 8 and 10, Mozilla Firefox 25, Apple Safari 6.1, Google Chrome 31.0.1650.57 that have been executed on all supported platforms from the set of MS Windows XP SP3, MS Windows 7 SP1, MS Windows 8, Apple MacOS 10.8.5, Ubuntu 12.04, CentOS 6.3, and openSUSE 12.2.

According to the results for the web browsers (cf. Table 3), the studied versions of the Firefox, Internet Explorer 10 and Chrome browsers can be detected based on autonomously issued queries, while Safari and Internet Explorer 8 cannot be detected precisely. Moreover, as shown in Table 4, regardless of the type of browser used, the majority of the operating systems can be detected due to characteristic DNS queries; only Windows XP and openSUSE do not issue characteristic queries in the background.

Table 3: DNS-based fingerprinting of common desktop web browsers; *emphasized hostnames* were observed for a single browser only and don't offer any web content intended for human users.

Browser	**Hostnames**
Firefox	*aus3.mozilla.org download.cdn.mozilla.net fhr.data.mozilla.com services.addons.mozilla.org versioncheck-bg.addons.mozilla.org versioncheck.addons.mozilla.org* addons.mozilla.org cache.pack.google.com download.mozilla.org [x].pack.google.com safebrowsing-cache.google.com safebrowsing.clients.google.com tools.google.com
IE 8	urs.microsoft.com
IE 10	*iecvlist.microsoft.com t.urs.microsoft.com* ctldl.windowsupdate.com mscrl.microsoft.com urs.microsoft.com www.bing.com
Safari	apis.google.com clients.l.google.com clients1.google.com safebrowsing-cache.google.com safebrowsing.clients.google.com ssl.gstatic.com www.google.com www.google.de www.gstatic.com
Chrome	*safebrowsing.google.com translate.googleapis.com [xxxxxxxxxx].[domain]* apis.google.com cache.pack.google.com clients[x].google.com [x].pack.google.com safebrowsing-cache.google.com safebrowsing.clients.google.com ssl.gstatic.com tools.google.com www.google.com www.google.de www.gstatic.com

Symbols: [x]: placeholder for varying numbers/strings; **[domain]:** placeholder for configured search domain.

Table 4: DNS-based fingerprinting of desktop operating systems; *emphasized hostnames* were observed for a single operating system only and don't offer any web content intended for human users.

Windows XP	
U	update.microsoft.com download.windowsupdate.com
T	time.windows.com
Windows 7	
U	*au.download.windowsupdate.com* update.microsoft.com download.windowsupdate.com ctldl.windowsupdate.com
T	time.windows.com
P	*watson.microsoft.com*
N	*ipv6.msftncsi.com* teredo.ipv6.microsoft.com www.msftncsi.com dns.msftncsi.com isatap.[domain] wpad.[domain]
M	*gadgets.live.com weather.service.msn.com money.service.msn.com*
Windows 8	
U	*au.v4.download.windowsupdate.com ds.download.windowsupdate.com* ctldl.windowsupdate.com *bg.v4.emdl.ws.microsoft.com fe[x].update.microsoft.com fe[x].ws.microsoft.com definitionupdates.microsoft.com spynet2.microsoft.com*
T	time.windows.com
P	*watson.telemetry.microsoft.com sqm.telemetry.microsoft.com*
N	teredo.ipv6.microsoft.com www.msftncsi.com dns.msftncsi.com isatap.[domain] wpad.[domain]
C	mscrl.microsoft.com crl.globalsign.net ocsp.verisign.com evsecure-ocsp.verisign.com evintl-ocsp.verisign.com
L	*clientconfig.passport.net* login.live.com go.microsoft.com
M	*ssw.live.com client.wns.windows.com appexbingfinance.trafficmanager.net appexbingweather.trafficmanager.net appexsports.trafficmanager.net appexdb[x].stb.s-msn.com de-de.appex-rf.msn.com finance.services.appex.bing.com financeweur[x].blob.appex.bing.com weather.tile.appex.bing.com*
MacOS X	
U	*swscan.apple.com swdist.apple.com swcdnlocator.apple.com su.itunes.apple.com* swcdn.apple.com r.mzstatic.com s.mzstatic.com metrics.mzstatic.com
T	*time.euro.apple.com*
P	*radarsubmissions.apple.com internalcheck.apple.com* securemetrics.apple.com
C	ocsp.apple.com ocsp.entrust.net ocsp.verisign.com EVIntl-ocsp.verisign.com EVSecure-ocsp.verisign.com SVRSecure-G3-aia.verisign.com
L	*identity.apple.com configuration.apple.com init.ess.apple.com init-p[x]md.apple.com* albert.apple.com
M	*p[x]-contacts.icloud.com p[x]-caldav.icloud.com p[x]-imap.mail.me.com [x].guzzoni-apple.com.akadns.net ax.init.itunes.apple.com a[x].phobos.apple.com keyvalueservice.icloud.com* [x]-courier.push.apple.com itunes.apple.com
Ubuntu	
U	*changelogs.ubuntu.com*
T	*ntp.ubuntu.com geoip.ubuntu.com*
P	*daisy.ubuntu.com*
L	*_https._tcp.fs.one.ubuntu.com fs-[x].one.ubuntu.com* one.ubuntu.com
CentOS	
U	*mirrorlist.centos.org*
T	*[x].centos.pool.ntp.org*
openSUSE	
U	download.opensuse.org opensuse-community.org

Symbols: [U]: polling for updates; **[T]:** time synchronization; **[P]:** problem reporting; **[N]:** discovery of network connectivity; **[C]:** certificate validation; **[L]:** activities right after log-on; **[M]:** miscellaneous background activity; **[x]:** placeholder for varying numbers/strings; **[domain]:** placeholder for configured search domain.

Drei Jahre Master Online Digitale Forensik: Ergebnisse und Erfahrungen*

Dominik Brodowski[1] Andreas Dewald[2] Felix C. Freiling[2]
Steve Kovács[3] Martin Rieger[3]

[1] LMU München, dominik.brodowski@jura.uni-muenchen.de
[2] FAU Erlangen, {felix.freiling|andreas.dewald}@cs.fau.de
[3] Hochschule Albstadt-Sigmaringen, {rieger|kovacs}@fh-albsig.de

Abstract: Der „Master Online Digitale Forensik“ ist ein berufsbegleitendes Studienangebot der Hochschule Albstadt-Sigmaringen, welches das zunehmend relevante Gebiet der IT-Forensik adressiert. Dieser Beitrag beschreibt die Entstehungsgeschichte und das Konzept dieses Studiengangs, der in Kooperation mit zwei Universitäten betrieben wird. Außerdem werden die Erkenntnisse und Erfahrungen der ersten vier Jahre zusammengefasst.

1 Einleitung

Um Straftaten im Cyberspace wirksam verfolgen zu können, müssen Spuren auf digitalen Geräten so gesichert und untersucht werden, dass sie als Beweismittel in einem Strafverfahren vor Gericht eingesetzt werden können. Die wissenschaftlich-methodischen Grundlagen hierfür bilden den Kern der forensischen Informatik [DF12]. Forensische Informatik bezeichnet die Anwendung wissenschaftlicher Methoden der Informatik auf Fragen der Rechtspflege. Damit verbunden sind vielfältige Fragen im weiteren Kontext von digitalen Ermittlungen, etwa nach geeigneten Ermittlungsansätzen in konkreten Fällen. Alle diese Fragen werden unter dem Oberbegriff „digitale Forensik“ zusammengefasst. Digitale Forensik beschäftigt sich also mit der Suche, Identifizierung, Sicherung und Analyse digitaler Spuren.

Allgemein besteht ein hoher Bedarf an Experten im Bereich digitale Forensik. Die Anforderungen an sie sind multidisziplinär und erstrecken sich zumindest über Teilmengen der Informatik und der Rechtswissenschaften. Ermittelnde Personen, vor allem aber auch IT-Sachverständige, müssen bei Untersuchungen planmäßig und strukturiert nach dem Stand der Wissenschaft vorgehen. Eine fundierte wissenschaftliche Ausbildung mag dazu nicht notwendig sein, erhöht aber deutlich die Qualität der Arbeitsergebnisse und erleichtert es, auf neue technische und rechtliche Entwicklungen zu reagieren. Dies gilt unabhängig davon, ob diese Ermittlungen im Auftrag von Behörden (Polizei, Staatsanwaltschaften) oder Unternehmen (interne Sicherheitsabteilungen, Innenrevision, Wirtschaftsprüfung) durch-

*Dieser Beitrag ist gewidmet Professor Dr. Joachim Vogel (1963–2013).

geführt werden.

Bis vor wenigen Jahren musste man ins Ausland gehen, um eine fundierte wissenschaftliche Ausbildung im Bereich digitale Forensik zu erwerben. Stellvertretend für eine Reihe von ähnlichen Studienabgeboten im angelsächsischen Raum seien hier der Master in „Forensic Computing and Cybercrime Investigation“ am University College Dublin [UCD] sowie der Master in “Digital Forensic Science” des Champlain College [Cha] in den USA genannt. Diese sind jedoch stark praxisorientiert ausgerichtet und in den angelsächsischen Rechtsraum eingebettet. Mit dem „Master Online Digitale Forensik“ (kurz Master Online) existiert seit 2010 auch in Deutschland ein Studiengang, der sich in seiner Konzeption vor allem durch vier Merkmale auszeichnet:

- Der Studiengang ist ein *Angebot für Berufstätige*, d.h. er ist berufsbegleitend studierbar.
- Auf technischem Gebiet orientiert sich der Studiengang am internationalen Stand der Forschung. Auf rechtlichem Gebiet liegt der Fokus auf dem *deutschen Rechtsraum* in seinen europäischen Bezügen.
- Der Studiengang verbindet die *praxisorientierten Ausbildungsziele* einer Fachhochschule mit den *forschungsorientierten Zielen* zweier Universitäten. Er ist ein Studiengang der Hochschule Albstadt-Sigmaringen, wird jedoch in Kooperation mit der Ludwigs-Maximilians-Universität München und der Friedrich-Alexander-Universität Erlangen-Nürnberg betrieben.
- Der Studiengang ist ein offenes *Angebot für Berufspraktiker* aus Behörden *und* Industrie, im Gegensatz zu bisherigen Angeboten, welche zumeist ausschließlich Mitarbeitern von Strafverfolgungsbehörden offen stehen.

Die genannten Merkmale des Studiengangs bilden mitunter ein Spannungsfeld, dessen Stärke von vornherein nicht absehbar war. Dies betrifft etwa die widersprüchlichen Ziele von Praxis- und Forschungsorientierung, die Kompatibilität von juristischen und informatischen Inhalten sowie die Vereinbarkeit von Berufstätigkeit und Studium. Da der erste (zum Wintersemester 2010/2011 aufgenommene) Jahrgang derzeit das Studium abschließt, besteht für die Autoren, die den Studiengang initiiert und aufgebaut haben, Bedarf an einer Bestandsaufnahme der gesammelten Erfahrungen, die mit diesem Beitrag dokumentiert werden sollen. Die bisher veröffentlichten vorläufigen Erkenntnisse [FK13] sollen hier vertieft und der wissenschaftlichen Fachgemeinschaft zur Verfügung gestellt werden. Nicht zuletzt der tragische Unfalltod des dritten professoralen Mitinitiators Joachim Vogel im August 2013 hat den Wunsch einer kritischen Standortbestimmung verstärkt.

Im folgenden Abschnitt 2 geben wir einen kurzen Abriss über die Geschichte, Studienform und die Gestaltung des Master Online. In Abschnitt 3 berichten wir über empirische Ergebnisse und gesammelte Erfahrungen mit dem Studiengang. Der abschließende Abschnitt 4 fasst die Erkenntnisse zusammen.

2 Hintergrund

Wir geben zunächst einen Überblick über die Kenndaten des 2010 eingerichteten Studiengangs und seine Entstehungsgeschichte. Anschließend folgt ein Kurzabriss über das ursprünglich geplante und das aktuelle Curriculum.

2.1 Kenndaten des Studiengangs und Entstehungsgeschichte

Der berufsbegleitend angelegte Studiengang führt zum Abschluss „Master of Science“ und umfasste ursprünglich einen Gesamtarbeitsaufwand (Workload) von 90 ECTS-Leistungspunkten (European Credit Transfer System). Der ursprüngliche Entwurf des Studienplans erstreckte sich über vier Semester zuzüglich der Master-Thesis. Von den Bewerbern wurden einschlägige IT-Kenntnisse erwartet, die ggf. in einem einsemestrigen Propädeutikum im Umfang von maximal 30 ECTS erworben werden konnten. Der reguläre Arbeitsaufwand pro Semester betrug 15 ECTS-Punkte, die auf drei Module mit jeweils 5 ECTS-Punkten aufgeteilt waren.

Im Verlauf der Akkreditierung wurde festgestellt, dass das Propädeutikum für Bewerber ohne Informatik-Vorkenntnisse berufsbegleitend nicht in einem Semester studierbar ist. Eine Möglichkeit, diesen Einwand zu heilen, hätte darin bestanden, von den Bewerbern einen ersten Studienabschluss in Informatik oder einem ähnlichen Fach zu fordern. Da man das Programm aber auch für Quereinsteiger und Polizisten offen halten wollte, wurde das Propädeutikum auf zwei Semester gestreckt und in das Studium integriert. Heute sieht der Master Online eine Regelstudienzeit von 5 Semestern plus Master-Thesis vor und umfasst einen Gesamtaufwand von 120 ECTS.

Die Einrichtung des Studiengangs wurde ermöglicht durch die großzügige Förderung der Landesstiftung Baden-Württemberg im Rahmen des Förderprogramms „Master Online“ (2009–2012) [MWFK08], damals noch für drei baden-württembergische Hochschulen: die Hochschule Albstadt-Sigmaringen, die Universität Mannheim und die Universität Tübingen. Durch den Wechsel zweier Professoren an andere Universitäten kam das aktuelle Projektkonsortium zustande. Im Rahmen des zweistufigen Auswahlverfahrens erfolgte die Einreichung der Projektskizze „Digitale Forensik“ am 12.09.2008. Nach positiver Begutachtung der Projektskizze reichte das Konsortium am 20.1.2009 den Vollantrag ein, der zum 1.4.2009 bewilligt wurde.

Zwar wurde der ursprüngliche Antrag bewilligt, dennoch enthielt der Förderbescheid zahlreiche Hinweise auf Bedenken der Gutachter, zu der wir Stellung nehmen mussten. Ein wesentliches Monitum war das vorgeschlagege Kooperationsmodell der Hochschulen. Es sollte durch klare Verträge sichergestellt werden, dass die Fortführung des Studiengangs auch bei Personalwechseln gewährleistet werden könnte. Außerdem wurde noch eine belastbare Studie zur Marktgängigkeit nachgefordert. Es wurde außerdem darauf hingewiesen, dass der Weiterbildungsaspekt des Studiengangs im Vordergrund zu stehen habe und nicht die Ausbildung wissenschaftlichen Nachwuchses.

Die weitere Entwicklung während der Förderphase wurde durch einen vom Ministerium fr

Wissenschaft, Forschung und Kunst Baden-Württemberg im Rahmen der Programmförderung eingesetzten Beirat begleitet. Dem Beirat gehörten Fachvertreter der Hochschulforschung und issenschaftlichen Weiterbildung an. Experten aus dem Gebiet der Digitale Forensik waren nicht vertreten. Der Studiengang wurde schlussendlich 2010 eingerichtet und Ende 2012 durch die Akkreditierungsagentur ACQUIN an der Hochschule Albstadt-Sigmaringen akkreditiert. Damit endete auch die Begleitung durch den Beirat. Es ist jedoch geplant, ein so genanntes "Advisory Board" bestehend aus Nachfragern aus Behörden und Unternehmen zu gründen. Das Advisory Board soll sicherstellen, dass aktuelle Entwicklungen im Themengebiet hinreichend berücksichtigt werden und der Praxisbezug der Studieninhalte gewährleistet ist.

2.2 Zulassungsvoraussetzungen

Um in den Master Online aufgenommen zu werden, müssen die üblichen Zulassungsvoraussetzungen erfüllt sein: Studieninteressierte müssen einen ersten berufsqualifizierenden Abschluss eines mindestens sechssemestrigen Studiums (mindestens 180 ECTS) an einer Universität, Hochschule oder Berufsakademie vorweisen. Außerdem sollten diese mindestens ein Jahr einschlägige Berufserfahrung nach Abschluss des vorhergehenden Studiums (beispielsweise auf den Gebieten der Informatik, der Rechtswissenschaften mit IT-Bezug oder der polizeilichen Strafverfolgung) nachweisen können.

Das Zulassungssemester des Studiengangs ist das Wintersemester eines jeden Jahres. Eine Möglichkeit zur Verkürzung der Studiendauer ergibt sich, wenn Studierende bereits einschlägige Studienleistungen in einem vorangegangenen Studium erbracht haben. Über die Anrechnung dieser Studienleistungen entscheidet der Prüfungsausschuss auf Antrag entsprechend der Studien- und Prüfungsordnung.

2.3 Studienkonzept

Das Studienkonzept des Master Online sieht eine feste Jahrgangsgröße von 30 Studierenden vor, die größtenteils gemeinsam die Semester durchschreiten. Als Studienform wurde das so genannte *blended learning* gewählt, also ein Fernstudium mit einem etwa 25%igen Präsenzanteil. Der Anteil des Fernstudiums kann in asynchrone und synchrone Lernphasen unterteilt werden. Als asynchrone Lehrinstrumente kommen Lehrbriefe (didaktisch aufbereitete Vorlesungsskripte), Übungsaufgaben, Projekte, Gruppenarbeiten und Gruppendiskussionen zum Einsatz. Synchrone Lernphasen finden entweder online statt in Form von gemeinsamen Web-Konferenzen, oder auch in physischer Präsenz am Wochenende. Pro Modul gibt es im Durchschnitt ein Präsenzwochenende, an dem sich Lehrende und Studierende physisch an einem der beteiligten Standorte treffen.

Die Kapazitätsgrenze von zurzeit maximal 35 Teilnehmern soll eine stets günstige Betreuungsrelation gewährleisten. Dies gilt beispielsweise fr die Betreuungskapazität im PC-Raum an Präsenzwochenenden ebenso wie im virtuellen Klassenzimmer und bei münd-

lichen Prüfungen. Es wäre zwar prinzipiell möglich die Personal- und Raumkapazität zu skalieren. Dies ist jedoch aufgrund des *blended learning*-Konzepts nicht so einfach möglich wie bei einem reinem Fernstudiengang.

Das Betreuungskonzept ist zweistufig. Den Hauptanteil der Betreuung erbringen dafür hauptamtlich eingestellte wissenschaftliche Mitarbeiter, die so genannten Tele-Tutoren. Die Dozenten, die im Nebenamt fungieren, sind verantwortlich für die Inhalte und die Qualität des jeweiligen Moduls. Sie übernehmen zudem die Beantwortung spezieller Fragen, die vom Tele-Tutor nicht beantwortet werden können. Das Studienkonzept sieht explizit vor, die Studierenden mit ihren reichhaltigen Praxiserfahrungen in die Gestaltung der Module zu integrieren, etwa dadurch, dass diese Fallbeispiele einbringen und die Prüfungsanforderungen mit den Praxisanforderungen vergleichen und kommentieren. Aus diesem Betreuungskonzept und der im Vergleich zu einem grundständigen Studium aufwändigeren Betreuung in den Präsenz- und online-basierten Veranstaltungen resultiert die Jahrgangsgröße von maximal 30 Studierenden.

2.4 Studienplan

Im Studienplan zeigt sich eine der spezifischen Eigenheiten des Studiengangs, durch die sie sich auch von bisherigen Angeboten im angelsächsischen Raum abhebt. Ziel des Studiengangs ist es, Studierenden vertiefte Fertigkeiten im Bereich der digitalen Forensik zu vermitteln (Praxiswissen) und sie in die Lage zu versetzen, ihr Methodenspektrum selbständig weiterentwickeln zu können (Forschungskompetenz, jedoch nicht primär die Ausbildung zum Wissenschaftler). Die Praxisorientierung liegt vor allem in den grundlegenden Informatikmodulen, die durch die Hochschule Albstadt-Sigmaringen angeboten werden. Die fortgeschrittenen technischen und juristischen Module werden durch die beiden beteiligten Universitäten beigesteuert.

Abbildung 1 gibt einen Überblick über den ursprünglichen Modulaufbau und die Gliederung der Module in die thematischen Säulen „Datenträger“, „Netzwerk“, „Methodik“ und „rechtlicher Rahmen“ sowie die verschiedenen Vertiefungsstufen „Basis“, „Vertiefung I“ und „Vertiefung II“. Die Grundlagenmodule werden nahezu ausschließlich von der Hochschule Albstadt-Sigmaringen angeboten.

2.5 Kosten/Finanzierung

Wie bereits oben erwähnt, wurde die Einrichtung des Studiengangs durch die Landesstiftung Baden-Württemberg finanziert. Seit Mai 2012 trägt sich der Studiengang vollständig durch die kostendeckenden Studiengebühren, die etwa EUR 15.000 für den gesamten Studiengang betragen. Der Großteil der Einnahmen wird für die Gehälter der Tele-Tutoren und für Dozentenhonorare ausgegeben. Ein Anteil von etwa 20% der Einnahmen wird für Studiengangsmanagement und Marketing aufgewendet.

	Master-Thesis (6. Semester)				
Säule/ Studienaufbau	Säule 1: Datenträger	Säule 2: Netzwerke	Säule 3: Methodik	Säule 4: Rechtlicher Rahmen	
Vertiefung II	Reverse Engineering (Erlangen, 5. Sem.)	Browser- und Anwendungsforensik (Erlangen, 5. Sem.)	Wirtschaftskriminalität (München, 5. Sem.)	Cyberkriminalität und Computerstrafprozessrecht (München, 4. Sem.)	
Vertiefung I	Datenträgerforensik (Albstadt-Sigmaringen, 4. Sem.)	Live Analyse (Erlangen, 4. Sem.)	Grundlagen Digitaler Forensik (Erlangen, 3. Sem.)	Cyberkriminalität und Computerstrafrecht (München, 3. Sem.)	
Basis & Einführung	Betriebssysteme (Albstadt-Sigmaringen, 2. Sem.)	Rechnernetze (Albstadt-Sigmaringen, 2. Sem.)	IT-Sicherheit (Albstadt-Sigmaringen, 3. Sem.)	Informationsrecht (Tübingen, 2. Sem.)	
Grundlagen (alle 1. Sem.)	Grundlagen der Script-Programmierung (Albstadt-Sigmaringen)	Internet Grundlagen (Albstadt-Sigmaringen)	Grundlagen der Programmierung (Albstadt-Sigmaringen)	Vernetzte Rechner und Anwendungen (Albstadt-Sigmaringen)	Einführung in die Informatik (Albstadt-Sigmaringen)

Abbildung 1: Ursprünglicher Studienplan des Master Online: Module mit zuständiger Hochschule und Angabe des Semesters, in dem das Modul vorgesehen ist.

3 Ergebnisse und Erfahrungen

Im folgenden Abschnitt gehen wir zunächst auf einige Zahlen zum Studiengang ein und berichten anschließend über unsere Erfahrungen. Viele der empirischen Ergebnisse entstammen einer Studie von Beck [Bec12] über verschiedene Aspekte des Master Online. Dazu wurden die ersten drei Jahrgänge mit Hilfe eines Online-Fragebogens im Oktober 2012 befragt. Von den damals 93 Studierenden wurden 36 Fragebögen vollständig beantwortet. Dies entspricht einer Teilnahmequote von 39%.

3.1 Ergebnisse

Wir gehen nun auf einige empirischen Ergebnisse des Studiengangs ein, etwa die Entwicklung der Studierendenzahlen, die vertretenen Berufsgruppen und die Abbruchquote.

3.1.1 Studierendenzahlen, vertretene Berufsgruppen, Altersstruktur

Die Entwicklung der Studierendenzahlen seit Beginn des Studiengangs ist in Tabelle 1 dargestellt. Mit dem Ausscheiden des ersten Jahrgangs im Frühjahr 2014 ist die nominelle Kapazität von $4 \cdot 30 = 120$ Studierenden mittlerweile fast voll ausgeschöpft.

Ob ein Auswahlverfahren nach der Zulassungssatzung durchgeführt wird, entscheidet die Anzahl zulassungsfähiger Kandidaten zum Bewerbungsstichtag. Jeder Bewerber muss ein Motivationsschreiben vorlegen, das die persönliche Studienmotivation darstellt. Ferner ist die berufliche Praxis durch Arbeitszeugnisse, dienstliche Beurteilungen oder Referenzschreiben des Arbeitgebers nachzuweisen. Bei fehlender berufspraktischer Erfahrung

	Anz. Studierende				
Jahrgang	WS 2010	WS 2011	WS 2012	WS 2013	12/2013
JG 1 (2010)	34	30	27	26	26
JG 2 (2011)		36	29	27	27
JG 3 (2012)			33	28	28
JG 4 (2013)				35	34
Summe					115

Tabelle 1: Entwicklung der Studierendenzahlen pro Jahrgang. Stichtag der Zählung ist der Beginn des jeweiligen Wintersemesters (WS), bzw. heute (Dezember 2013).

muss der Kandidat einen Eignungstest bestehen, um zum Studium zugelassen zu werden. Bisher wurde kein Auswahlverfahren mit Studienplatzvergabe nach Ranking durchgeführt. Mit Bewerbern jenseits der Kapazitätsgrenze wurde vereinbart, das Studium im Folgejahr zu beginnen.

Die Entwicklung der Studierendenzahlen pro Jahrgang zeigt, dass trotz strenger Zulassungskriterien das erste Studienjahr eine signifikante Hürde für die Studierenden darstellt. Nach detaillierter Betrachtung aller 23 bisherigen Studienabbrecher wird deutlich, dass 13 von ihnen bereits während oder am Ende des ersten Semesters das Studium beenden. Die Gründe hierfür wurden bisher nicht systematisch erhoben. Eine nicht-repräsentative Umfrage unter drei Studienabbrechern aus dem Sommersemester 2013 brachte das Ergebnis, dass bei zwei von Ihnen die hohe Belastung durch Beruf, Familie und Studium Ursache für den Abbruch war. Es sind aber aus vorherigen Jahrgängen auch Fälle bekannt, in denen gesundheitliche Probleme den Grund für den Studienabbruch darstellten.

Tabelle 2 schlüsselt die Studierendenzahlen der verschiedenen Jahrgängen nach Geschlechtern auf. Zum Zeitpunkt der jeweiligen Einschreibung waren in der Summe über alle Jahrgänge 13 von 138 Studierenden weiblich (9%). Zum aktuellen Zeitpunkt (Dezember 2013) sind über alle Jahrgänge hinweg 12 von 115 Studierenden weiblich (10%). Diese Geschlechterverteilung entspricht in etwa denen von Präsenzstudiengängen im Bereich Informatik. Die Tabelle enthält auch Angaben zu den Abbrecherquoten pro Jahrgang. Da bisher nur ein Jahrgang kurz vor dem Abschluss steht, sind diese Zahlen natürlich schwer vergleichbar. Die Abbruchquote insgesamt liegt bei 16% und damit deutlich niedriger als bei Präsenzstudiengängen im Fach Informatik. Es zeigt sich auch, dass die Abbrecherquote der weiblichen Studierenden deutlich niedriger ist als die der männlichen Studierenden (über alle Jahrgänge hinweg 7 statt 17%).

Die im Studiengang vertretenen Berufsgruppen sind in Abb. 2 dargestellt. Grundlage für die Übersicht sind 116 der 117 zum Beginn des Wintersemester 2013/2014 aktiven Studierenden. Aus der Übersicht wird deutlich, dass mehr als die Hälfte der Studierenden (62) aus Behörden stammen, die Mehrheit davon (33) aus Polizeibehörden. Von diesen sind die Landeskriminalämter mit 19 Personen am stärksten vertreten. Hier zeigt sich, dass der Zuschnitt und die Zielrichtung des Studiengangs offenbar für Spezialisten in Fachabteilungen besonders attraktiv erscheint.

Beck [Bec12] erhob auch die Altersstruktur der Studierenden, die bisher nicht explizit als Wert registriert wurde. Die erhobenen Daten sind in Abb. 3 dargestellt. Für die Darstellung

	Studierendenzahlen zum Zeitpunkt								
	Einschreibung			heute (12/2013)			Abbr.quote in %		
Jahrgang	m	w	ges.	m	w	ges.	m	w	ges.
JG 1 (2010)	28	6	34	21	5	26	25	16	23
JG 2 (2011)	35	1	36	26	1	27	26	0	25
JG 3 (2012)	31	2	33	26	2	28	16	0	15
JG 4 (2013)	31	4	35	30	4	34	3	0	2
Summe	125	13	138	103	12	115	17	7	16

Tabelle 2: Studierendenzahlen und Abbrecherquote nach Geschlecht.

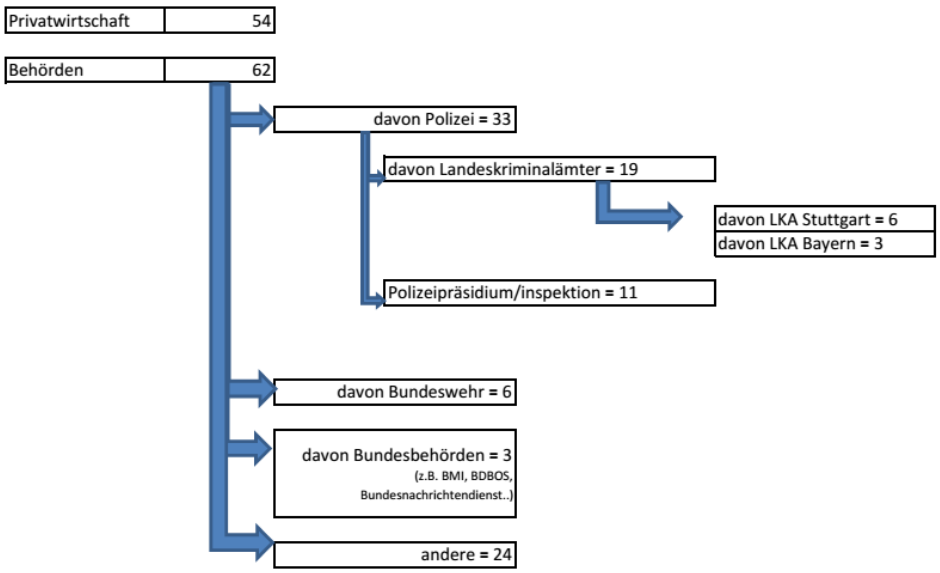


Abbildung 2: Berufsgruppen über alle aktuellen Studierenden.

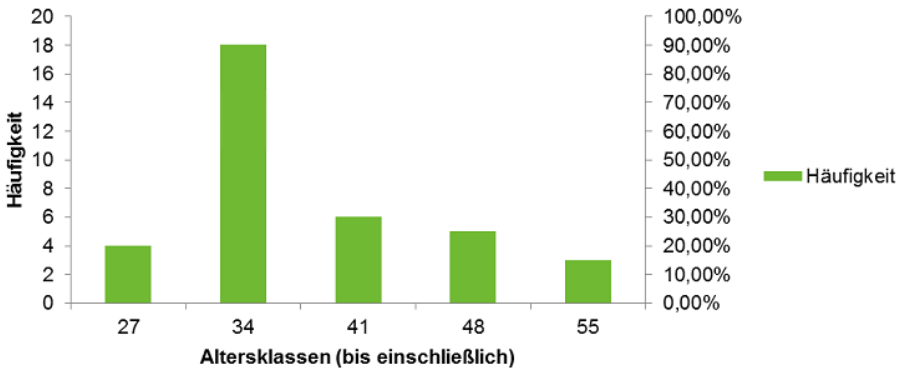


Abbildung 3: Alterstruktur der Teilnehmer nach Altersklassen [Bec12, S. 58].

wurden fünf Altersklassen zu jeweils sieben Jahren gebildet. Die erste Klasse erfasste alle Teilnehmer zwischen 20 und 27 Jahren, die zweite diejenigen zwischen 28 und 34 Jahren usw. Die Mehrheit der Teilnehmer ist der Alterklasse 28–34 zugeordnet. Die Spannweite des erhobenen Lebensalters lag bei 33 Jahren (22–33 Jahre). Der Median liegt ebenfalls bei 33 Jahren.

3.1.2 Finanzierung/Unterstützung durch den Arbeitgeber

Für Aussenstehende meist überraschend sind die Ergebnisse bezüglich der Unterstützung der Studierenden durch den Arbeitgeber. Da die Studierenden einen nicht unerheblichen Zeitaufwand in das Studium investieren, würde man erwarten, dass sie nicht auch noch zusätzlich bereit sind, die vollen Kosten des Studiums zu übernehmen, zumal die Weiterqualifikation in vielen Fällen auch dem Arbeitgeber zugute kommt. Die Ergebnisse von Beck [Bec12, S. 59f] zeigen jedoch, dass die überwiegende Mehrheit der Teilnehmer das Studium selbst finanziert.

Von 36 Studierenden erhalten 25 weder eine finanzielle noch eine zeitliche Unterstützung durch ihren Arbeitgeber. Die verbleibenden 11 Studierenden werden alle von ihrem Arbeitgeber finanziell unterstützt, zwei davon zusätzlich noch durch eine zeitweise Freistellung von der Arbeitszeit. Laut Beck mussten drei Personen die Studiengebühren nicht zu 100% selbst zahlen, sondern es wurden Teile der Studiengebühren durch den Arbeitgeber übernommen. Zu diesen drei Personen zählen auch die beiden Studierenden, die eine zeitliche Unterstützung durch den Arbeitgeber erhielten. Beim Großteil der Studierenden lag der Anteil der durch den Arbeitgeber übernommenen Studiengebühren bei 50–75%.

3.1.3 Studierendenzufriedenheit

Die Evaluation von Beck [Bec12] untersuchte auch zusammenfassend die Zufriedenheit der Studierenden mit dem Studienangebot.

Auf die Aussage „Das Studium entspricht voll und ganz meinen Vorstellungen“ antworteten von 35 Studierenden 22 (also 63%) entweder mit „stimme voll zu“ (3) oder „stimme eher zu“ (19) [Bec12, S. 72]. Ein wichtiger Indikator für die Zufriedenheit ist auch die Antwort auf die Frage, ob die Studierenden das Studium weiterempfehlen würden. Hier antworteten 22% (8) mit vorbehaltloser Zustimmung, weitere 16 (44%) mit bedingter Zustimmung [Bec12, S. 72].

3.2 Erfahrungen

Neben den „objektiven“ Zahlen möchten wir auch noch subjektive Erfahrungen der Autoren in die Diskussion einbringen, die wir grob in drei Bereiche gegliedert haben.

3.2.1 Erfahrungen mit berufsbegleitendem Studium

Vor der Einrichtung des Studiengangs hatte keiner der Autoren Erfahrungen mit der Durchführung eines berufsbegleitenden Studiums. Die Erfahrungen mit dieser Studienform nach drei Jahren sind tendentiell positiv, verlangen aber eine differenzierte Darstellung.

Zunächst war der Einrichtungsaufwand sowohl personell als auch inhaltlich beträchtlich. So mussten die Fernstudienmaterialien grundlegend neu erarbeitet und auf die Bedürfnisse Berufstätiger zugeschnitten werden. Der Fachkräftemangel, der durch das Projekt eigentlich bekämpft werden soll, führte dazu, dass Personalstellen teilweise nur mit großer Verzögerung besetzt werden konnten. Die mangelnde Erfahrung von Dozenten und Tutoren mit der neuen Studienform erzeugte zuerst vielfach Frustation bei den Studierenden über die technischen und juristischen Module (Berührungsängste, hohe Schwierigkeit, nicht geringer Arbeitsaufwand), die sich mit der Zeit dann doch in eine gewisse Motivation wandelte.

Rundherum positiv war die Tatsache, dass die Studierenden als Berufspraktiker ihre eigenen Erfahrungen mitbringen und die Inhalte leichter in den Berufsalltag einordnen können als Präsenzstudenten. Die Möglichkeiten solcher Verknüpfung hängen jedoch sehr stark vom konkreten Berufsumfeld ab. Die Berufspraktiker sind auch deutlich kritischer als Präsenzstudierende: Sie hinterfragen den Sinn und Zweck der vorgestellten Konzepte viel stärker, was die Dozenten im Gegenzug zu einer intensiveren Vorbereitung zwingt. Allerdings müssen Berufspraktiker häufig erst wieder das „Lernen lernen“, etwa relevante Inhalte und Konzepte selbst zu erkennen und darauf zu fokussieren. Auch selbstständige und kreative Arbeitsweise ohne exakte Arbeitsanweisungen und -vorgaben müssen anfangs erst einmal wieder trainiert werden.

Eine relevante Qualifikation, die im Master Online quasi nur nebenbei vermittelt wird, ist das Verfassen wissenschaftlicher Arbeiten. In Präsenzstudiengängen erfolgt dies meist in Seminaren. Im Master Online beschränkt sich der Kompetenzerwerb im Schreiben (und der mündlichen Präsentation/Verteidigung) forensischer Berichte und juristischer Hausarbeiten, welche natürlich auch Mindestanforderungen an wissenschafltiches Arbeiten erfüllen müssen.

Allgemein erscheinen die berufstätigen Studierenden im Durchschnitt sehr viel zielorientierter und motivierter als die Präsenzstudierenden. Sie stehen allerdings auch unter hohem Zeit- und Erfolgsdruck (Arbeitgeber, eigene Familie, Karriere). Beck beobachtete etwa, dass die zusätzlichen finanziellen Belastungen etwa im Kontext von Präsenzwochenenden hinter der effizienten Nutzung der Präsenzstunden regelmäßig abfielen: „Dies bedeutet, dass Zeit als Ressource subjektiv höher bewertet wird als Geld." [Bec12, S. 63] Wir beobachten bei vielen Studierenden, dass berufliche und teilweise auch familiäre Erfordernisse den Studienfortschritt begrenzen. So kommt es des öfteren vor, dass Studierende einen Präsenztermin wegen dienstlicher Aufgaben nicht wahrnehmen können oder einen Prüfungstermin wegen einer Dienstreise in einem Nachholtermin absolvieren müssen. Bei längerfristigerer Absorption der „Freizeit" und Mobilität durch berufsbedingte Auslandsaufenthalte oder durch eine Familienphase beantragen Studierende ein oder mehrere Urlaubssemester. Dadurch und durch eine dann anfallende Phase des Wiedereinstiegs verlängert sich bei diesen Studierenden das Studium. Die Studienorganisation reagiert auf die beruflichen und persönlichen Umstände flexibel, indem sie zusätzliche Prüfungstermine anbietet und alternative Möglichkeiten in Studienberatungen aufzeigt.

Die Herausforderung besteht für Dozenten, Tutoren und Studierende, im Spannungsfeld zwischen Zeitdruck, Erfolgsdruck und Qualitätsansprüchen mit der richtigen Didaktik den optimalen Arbeitspunkt zu finden. Es kam häufig vor, dass Anforderungen, die wir als Dozenten stellten, zum Teil als nicht berufsbegleitend studierbar empfunden wurden. Es ist eine immer wieder neue Herausforderung, den Studierenden Flexibilität zu erlauben ohne Einbußen in der Qualität des Lehrangebotes hinzunehmen. Dies erfordert eine ständige Anpassung und Überprüfung der Inhalte und der didaktischen Konzepte.

3.2.2 Erwartungshaltung und Praxisbezug

Der Spagat zwischen berufspraktischer und forschungsnaher Ausbildung ist uns nur zum Teil gelungen. Hier müssen wir trotz umfangreicher Anstrengungen selbstkritisch anmerken, dass wir mehr tun müssen, um auf die spezifischen Belange von Berufstätigen einzugehen. Gerade für den forschungsorientierten Ansatz müssen die Module mit einem durchgehenden „roten Faden" versehen sein und Fragestellungen jeweils aus der Praxis motiviert werden. Auch die motivierende, durchgängige und service-orientierte Betreuung muss verbessert werden. Berufstätige benötigen Betreuung vor allem abends und an Wochenenden, Zeiten also, an denen Dozenten in Präsenzstudiengängen selten für Studierende zur Verfügung stehen.

Wie eingangs geschildert, entwickelt sich das Fach der digitalen Forensik mit der technischen Entwicklung stürmisch weiter. Konkretes technisches Wissen, wo Spuren zu finden sind und was sie bedeuten, veraltet also sehr schnell. Der Studiengang versucht, auf diese Entwicklung dadurch zu reagieren, dass aktuell relevante Technologien und Fragestellungen in den Kursen besprochen werden. Aber gerade in diesem Punkt kommen die Vorzüge der forschungsorientierten Herangehensweise zum Tragen: Die Studierenden werden in die Lage versetzt, selbständig mit der neuen Entwicklung Schritt zu halten, indem sie lernen, selbständig neue Spurenquellen zu erschliessen und ihre Bedeutung einzuschätzen. Dies führt mitunter zu enttäuschten Erwartungen der Studierenden, die gehofft hatten, zu

„digitalen Ermittlern“ ausgebildet zu werden, die sofort einen Fall bearbeiten und aufklären können. Hierzu benötigt man allerdings zusätzlichen kriminalistischen Sachverstand, den allenfalls die Studierenden mit Hintergrund in der Strafverfolgung in ausreichendem Maße mitbringen. Dies war bei der Konzeption des Studiengangs unklar und spiegelt eine wichtige Entwicklung im Bereich der digitalen Forensik wider: Die zunehmende Trennung von forensischen Sachverständigen, die Spurenquellen erschließen und deren Bedeutung interpretieren können, und digitalen Ermittlern, die Spuren auswerten. Über eine Stärkung dieses „ermittelnden“ Kompetenzbereichs im Themenspektrum des Masterstudiengangs muss also nachgedacht werden.

3.2.3 Fachspezifische Erfahrungen

Neben den eben geschilderten allgemeinen Erkenntnissen gibt es noch eine Reihe von Erfahrungen, die spezifisch sind für das Fach der digitalen Forensik.

Ein besonders positiver Punkt war die Erkenntnis, dass mündliche Prüfungen hervorragend genutzt werden können, um die Präsentation und Verteidigung von forensischen Berichten zu üben. Die Prüfer nehmen dabei explizit die Rolle des Richters, Staatsanwalts oder eines gegnerischen Sachverständigen ein und simulieren eine Gerichtsverhandlung. Je nach Engagement der Prüfer stellt dies eine sinnvolle, praxisnahe und zielführende Prüfungsform dar, in der der eigentlich versteckte Lehrplan vorab explizit gemacht werden kann. So wird von vornherein darauf hingewiesen, dass Suggestivfragen gestellt werden können, um Gutachter zu verunsichern, wie es auch häufig in der Praxis geschieht.

Subjektiv ist zu beobachten, dass die Studierenden in der Tat zum Ende des Studiums in der Lage sind, selbstständig vorgegebene Fälle zu bearbeiten und die Untersuchungsergebnisse überzeugend zu erläutern. Die Kompetenzen korrellieren aber deutlich mit den Vorkenntnissen. Das Studienziel, forschungsnah arbeiten zu können und durch Anwendung wissenschafltlicher Methoden der Informatik neue Spurenquellen zu erschliessen und deren Bedeutung objektiv zu bewerten, haben wir bei einem Großteil der Studierenden des ersten Jahrgangs erreicht. Unsere Absolventen lassen sich zum großen Teil nicht (mehr) durch die Funktionalität vorhandener Werkzeuge einschränken sondern sind in der Lage, neue Wege zu gehen und die Entwicklung der forensischen Informatik voranzutreiben. In Verbindung mit den juristischen Kompetenzen entsteht ein durchaus einmaliges Kompetenzprofil, das sich allerdings in der Praxis nun bewähren muss. Defizite liegen, wie bereits oben besprochen, im Wissen über konkrete und im Einzelfall eingesetzte (kommerzielle) Technologien und Ermittlungstaktiken.

4 Schlussfolgerungen und Zusammenfassung

Nach drei Jahren Erfahrung mit dem Master Online haben wir uns die Frage gestellt, welche grundsätzlichen Einsichten langfristig auch für andere Wissenschaftler und Bildungsplaner wertvoll sein könnten. Wir haben die wichtigsten Einsichten in den folgenden Absätzen zusammengefasst.

Fördergelder sind notwendig. Da der Einrichtungsaufwand für berufsbegleitende Studiengänge außerordentlich hoch ist, müssen dafür hinreichend viele Ressourcen bereit gestellt werden. Für uns waren dies die Fördergelder der Landesstiftung Baden-Württemberg, ohne die der Studiengang nicht existieren würde. Da ein Studienkonzept selten im „ersten Wurf“ gelingt, sollte in zukünftigen Förderprogrammen die Möglichkeit vorgesehen werden, die Studienkonzepte im Rahmen von „Pilotphasen“ an echten Studierenden zu testen, die im Gegenzug reduzierte Studiengebühren bezahlen. Dies scheint in aktuellen Förderprogrammen wie „Offene Hochschulen“ [BMBF13] des BMBF bereits berücksichtigt worden zu sein.

Nichts ist praktischer als eine gute Theorie. Berufsbegleitendes Studium zwingt die Hochschulen (vor allem die Universitäten), die Relevanz ihrer Forschungs- und Lehrinhalte für die Praxis ständig zu überprüfen. Gleichzeitig bringt berufsbegleitendes Studium die Studierenden dadurch zur nachhaltig relevanten Theorie. Übungen und Fallbeispiele müssen einen konkreten Praxisbezug aufweisen, damit sie von den Studierenden als sinnvoll akzeptiert werden. Um dies zu erreichen, müssen die Hochschulen eng mit den Bedarfsträgern (Ermittlungsbehörden und Industrie) kooperieren. Die ursprüngliche Planung, dass die Fachhochschule den Praxisbezug einbringt und die Universitäten die Forschungsorientierung, ist in dieser Reinform nicht sinnvoll. Die genauen Beiträge, die die verschiedenen Partner zu den erworbenen Kompetenzen beisteuern, ist jedoch noch unklar und muss evaluiert werden.

Durchgängiger Themenbezug. Im aktuellen Curriculum werden forensische Studieninhalte zu spät vermittelt. Das Curriculum wird darum in Zukunft dahingehend verändert, dass das einführende Forensik-Modul („Grundlagen Digitaler Forensik“) im Studienplan vorgezogen wird. Alle Module müssen mit Bezügen zum Oberthema des Studiengangs ausgestattet werden. Die zunehmende Breite des Gebietes soll durch einen Wahlpflichtbereich adressiert werden. Die geplanten Änderungen des Curriculums sind in Abb. 4 dargestellt.

Auf Zeiteffizienz hin optimieren. Zeitknappheit bei den Studierenden beeinflusst maßgeblich den Studienerfolg und das Studienerlebnis. Die Zeiteffizienz ist also der kritischste Faktor, auf den hin optimiert werden muss. Dies muss nicht der traditionellen Humboldt'schen Aufassung eines Studiums in „Einsamkeit und Freiheit“ widersprechen, denn berufsbegleitende Studierende agieren *per se* durch den hohen Eigenstudiumsanteil in größerer „Einsamkeit“ als Präsenzstudierende. Der Aufbau der Studienmodule muss aber durchgängig die Zeiteffizienz der Lehr- und Lernmethoden berücksichtigen. Dies sollte in Zukunft stärker erfasst und evaluiert werden, um besser zu verstehen, welche Lehr- und Lernformen für welche Arten von Stoff geeignet sind.

In diesem Kontext sollte den Studierenden auch der Blick für das richtige Verhältnis aus Eigenstudium und Erholung geschärft werden. So sollte es Studierenden auch ermöglicht werden, die Studienlast auch über die Semester hinweg zu verteilen, also beispielsweise in einem Semester nur ein oder zwei Module zu belegen und die anderen später nach-

	Master-Thesis (7. Semester)			
Säule	**Säule 1: Rechnersysteme**	**Säule 2: Vernetzung**	**Säule 3: Methodik + Wahlpflicht**	**Säule 4: Rechtlicher Rahmen**
Vertiefung	Reverse Engineering (Erlangen, 4. Semester)	Browser- und Anwendungsforensik (Erlangen, 6. Semester)	Wahlpflichtmodul (gemäß Modulhandbuch, 6. Semester)	Wirtschaftskriminalität (München, 6. Semester)
	Datenträger-Forensik (Albstadt-Sigmaringen., 4. Semester)	Live Analyse (Erlangen, 5. Semester)	Digitale Ermittlungen (Albstadt-Sigmaringen, 5. Semester)	Cyberkriminalität und Computerstrafprozessrecht (München, 5. Semester)
	Betriebssysteme und Betriebssystemforensik (Alb.-Sig., 3. Semester.)	Rechnernetze und Netzwerkforensik (Alb.-Sig., 3. Semester)	Grundlagen Digitaler Forensik (Erlangen, 2. Semester)	Cyberkriminalität und Computerstrafrecht (München, 4. Semester)
Grundlagen II	IT-Sicherheit und IT-Angriffe (Alb.-Sig., 2. Semester)	Programmieren im Forensik-Umfeld (Alb.-Sig., 2. Semester)	Informationsrecht (Tübingen, 3. Semester)	
Grundlagen I	Einführung in die Informatik (Albstadt-Sigmaringen, 1. Sem.)	Einführung in Betriebssysteme und Methoden der Informatik (Albstadt-Sigmaringen, 1. Semester)	Internet Grundlagen (Albstadt-Sigmaringen, 1. Semester)	

Abbildung 4: Neues Curriculum des Master Online Digitale Forensik, gültig ab dem Sommersemester 2014. Neben der Einführung eines neuen Moduls „Digitale Ermittlung“ und eines Wahlpflichtbereichs wurden das einführende Forensik-Modul im Studienplan vorverlegt und die anspruchsvollen technischen Module gleichmäßiger auf die Semester verteilt. Außerdem wurden die Grundlagenkurse des ersten Semesters auf zwei Semester verteilt.

zuholen. Durch einen rigiden, auf Jahreskohorten ausgerichteten Studienplan wird diese Flexibilität stark eingeschränkt. Die Abbrecherquote ist trotz all dieser Einschränkungen deutlich niedriger als im Präsenzstudium. Die konkreten Gründe hierfür müssen weiter untersucht werden.

Programmmanagement kostet Geld. Da der Studiengang aus kostendeckenden Studiengebühren finanziert wird, kann dieser nur so lange existieren, wie eine Nachfrage nach Studienplätzen existiert. Dies ist uns nur durch intensives Marketing und ein engagiertes Programmmanagement gelungen. Hierfür müssen auch Kosten eingeplant werden. Der buchhalterische Anteil dieses Bereichs an den Gesamteinnahmen liegt im Master Online bei etwa 20% und erscheint bei den vielfältigen Aufgaben der Studierendenbetreuung als recht knapp. Der Anteil kann nur dann konstant gehalten werden, wenn eine effiziente IT-Infrastruktur zur Studierenden- und Moduladministration zur Verfügung steht.

Danksagungen

Wir danken den Reviewern und Marion Liegl für Kommentare zu diesem Text sowie Franziska Bantle für die Zusammenstellung der in diesem Beitrag verwendeten empirischen Daten.

Literatur

[Bec12] Carolin Beck. Konzeption von E-Learninginhalten unter Berücksichtigung von lerntheoretischen Grundlagen und Kundenwünschen am Beispiel eines berufsbegleitenden onlinebasierten Masterstudiengangs. Bachelorarbeit, Hochschule Albstadt-Sigmaringen, Dezember 2012.

[BMBF13] Bundesministerium für Bildung und Forschung. Wettbewerb "Aufstieg durch Bildung: offene Hochschulen". `http://www.bmbf.de/de/17592.php`, Dezember 2013.

[Cha] Champlain College. Master of Science in Digital Forensics Science. online: `http://www.champlain.edu/computer-forensics/masters-digital-forensics-science`. letzter Zugriff: 12.2.2014.

[DF12] Andreas Dewald und Felix C. Freiling. *Forensische Informatik*. Books on Demand, 2012.

[FK13] Felix Freiling und Steve Kovacs. Master Digitale Forensik: Erste Erfahrungen. *digma – Zeitschrift für Datenrecht und Informationssicherheit*, 13(1):38–41, März 2013.

[MWFK08] Ministerium für Wissenschaft, Forschung und Kunst Baden-Württemberg. Bekanntmachung des Ministeriums für Wissenschaft, Forschung und Kunst Baden-Württemberg über die Ausschreibung zur Einrichtung online-gestützter Weiterbildungsstudiengänge – Förderprogramm „Master Online", 10. Januar 2008. Az.: 34-802.67/141.

[UCD] University College Dublin. MSc Forensic Computing and Cybercrime Investigation. online: `http://www.csi.ucd.ie/content/msc-forensic-computing-and-cybercrime-investigation`. letzter Zugriff: 12.2.2014.

Towards Suppressing Attacks on and Improving Resilience of Building Automation Systems - an Approach Exemplified Using BACnet

Sebastian Szlósarczyk[1], Steffen Wendzel[1], Jaspreet Kaur[1],
Michael Meier[1], Frank Schubert[2]

[1] Fraunhofer FKIE, Bonn, {szlos,wendzel,kaur,meier}@cs.uni-bonn.de
[2] MBS GmbH, Krefeld, frank.schubert@mbs-software.de

Abstract: Different concepts of IT security, like communication encryption, have already been applied to building automation systems (BAS). However, no research is available to mitigate malicious or incompliant network traffic in BAS. Both aspects are covered by traffic normalizers. We present the first work-in-progress research on traffic normalization for building automation networks exemplified using *building automation control and network* (BACnet) protocol.

1 Introduction

Building automation systems (BAS) are IT components integrated in and capable to control and monitor buildings. BAS are aiming to improve the energy efficiency of houses, to increase the comfort and safety for people living or working in a building, and to decrease a building's operation costs. Therefore, it is necessary to enable a BAS to control critical equipment like smoke detectors or physical access control components.

BAS form networks which can be interconnected with other buildings and the Internet (e.g., for remote monitoring purposes) and therefore use different protocols, especially the *building automation control and network* (BACnet), the *European Installation Bus* (EIB)/*Konnex* (KNX), and the *Local Operating Network* (LON). These protocols are linked to security features specified in their standards, which were improved over the time. However, even if security features are available in standards they are commonly not integrated in devices or used in practice.

In this paper, we focus on BACnet, an open data communication protocol developed by ASHRAE (*American Society of Heating, Refrigerating and Air Conditioning Engineers*). BACnet is standardized by ANSI standard 135 and ISO standard 16-484 part 5 and 6, it is integrated into products by more than 700 vendors worldwide [Bac13].

A means to improve network reliability and security are *traffic normalizers* (also known as *protocol scrubbers* [MW+00] and from TCP/IP networks). Traffic normalization is not available for *any* BAS protocol. A traffic normalizer actively modifies or drops network traffic in order to remove potentially malicious or incompliant elements regarding to a standard, i.e., a traffic normalizer is an "*active mechanism that explicitly*

removes ambiguities from [...] network flows" [MW+00]. Therefore, traffic normalizers analyze header elements of network packets and drop malicious packets or clear/modify suspicious bits in header fields before a network packet is forwarded. In TCP/IP networks, traffic normalization is capable to prevent various attacks on TCP/IP stacks.

As pointed out by Bowers, *BACnet devices are not robust enough to deal with abnormal traffic* [Bow13] as protocol implementations are vulnerable against malformed packets and various forms of attacks. Such attacks can, for instance, cause a denial of service in smoke detectors or other critical BAS equipment. Due to the Internet connectivity of BACnet systems and the fact that BACnet devices can be found using the *SHODAN* search engine (cf. *www.shodanhq.com*), various attacks on BACnet are easy to perform.

We present the first research on the feasibility of traffic normalization for building automation systems and exemplify our work using the BACnet network layer. We lay the foundation for a Snort-based proof-of-concept implementation which we currently develop. We design our normalization to be capable to significantly increase the robustness of BAS networks by protecting BACnet network stack implementations against malformed packets and packets linked to selected attacks as well as by ensuring the compliance of BACnet messages. Our normalization rules are additionally a means to counter *fuzzing* attacks and to provide protection for usually seldomly updated BACnet devices as patching is a challenging task in BAS. Our future work also investigates normalization options for the BACnet application layer [ASH13].

Our system is designed to normalize traffic between BAS subnets (e.g., between different floors of a building or between separated buildings). The implementation of our normalizer assures that the transferred packets reach the *receiver* well-formed according to the protocol standard.

We moreover analyze the BACnet network layer with its potential vulnerabilities and its potential incompliant behavior. On the one hand, we examine the ISO DIN Norm 16484-5 in order to rule out packets of wrong format [ND12a]. On the other hand, well-known attacks are adopted from IP to the corresponding BACnet network layer. In addition, potential exploitation of so called *network protocol data unitsmal* (NPDUs) is specified. Within this examination, we emphasize the different network messages and the importance of the a*pplication protocol data unit* (APDU) segmentation.

Hereafter the paper is organized as follows. We discuss related work in Section 2. Section 3 explains the BACnet network layer and Section 4 presents potential attacks. We introduce normalization rules deduced from these attacks as well as normalization rules to ensure compliance in Section 5 and conclude in Section 6.

2 Related Work

Former BAS were designed to work as isolated stand-alone systems with basically no security features. Due to the need to increase BAS' functionality, inter-connectivity, inter-operability, and especially Internet access for BAS became significant features.

However, the inter-connectivity of BAS enables remote attacks. Attacks on BAS can focus on getting physical access to a building by exploiting window or door actuators [Hol03], on getting access to an organizational intranet [SZ12], on terrorist attacks (e.g., turning off fire alarms before a fire is placed) [Hol03], on monitoring inhabitants [WKR12], or on disabling a building's functionality via DoS attacks [GKNP06].

Recently, the *BACnet Attack Framework* was presented [Bow13], which highlights selected attack techniques, including attacks on BACnet routing. The *BACnet Firewall Router* (BFR) is the first approach to integrate simple firewall functionality in BACnet [HBG06]. The BFR firewall is an open source project that implements filters for BACnet messages and is capable to realize NAT, software-realized switching, and routing.

In contrast to network intrusion detection systems (NIDS), a traffic normalizer actively modifies network traffic instead of passively monitoring it to detect malicious elements. Moreover, normalizers can be used to modify traffic before it reaches a NIDS in order to protect the NIDS [MW+00]. In comparison to firewalls which only drop traffic, normalizers can additionally modify the traffic. Today, many firewalls (e.g. OpenBSD *pf*) and NIDS (e.g., Snort's *inline mode*) support traffic normalization. The BFR does not comprise any normalization capabilities.

Closely related to this work is our own previous work on the elimination of covert channels using a traffic normalizer to enforce multi-level security (MLS) for BACnet [Wen12a,WKR12]. We moreover presented a protocol-independent middleware that ensures MLS to prevent the monitoring of inhabitants as well as it normalizes actuator commands for BAS (e.g., preventing to raise the temperature in a room to 28°C if only 22°C is allowed) [Wen12b]. In comparison to previous work, we present the first traffic normalization for BAS capable *i)* to counter typical attacks known from TCP/IP networks, *ii)* to ensure compliance, and *iii)* to increase robustness against vulnerability tests and fuzzing attacks.

3 BACnet in a Nutshell

BACnet is used in BAS to handle a number of application areas such as heating, ventilation, and air-conditioning (HVAC), lighting, fire alarming etc. in a cost effective and reliable manner [Tom12]. BACnet defines four layers (physical, data link, network, and application layer) similar to the particular functions of the OSI layers (visualized in Fig. 1) but with the special difference that the BACnet application layer is additionally responsible for performing and handling message segmentation/reassembly. The maximum length of messages shall not exceed the capability of any data link technology encountered along the path from source to destination.

A unified, data link layer-independent communication takes place at the BACnet network and application layer but various technologies can be used at lower levels. BACnet network layer messages can, for instance, be encapsulated in UDP (referred to as BACnet/IP), in the BACnet-specific protocol MS/TP (RS485) and in LonTalk Zigbee.

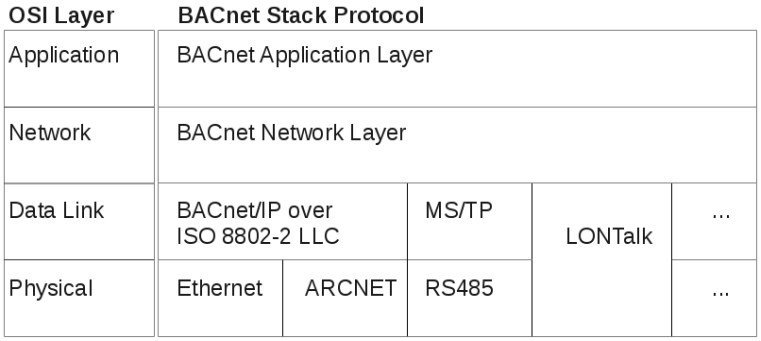


Figure 1: BACnet OSI Layers

As our aim is to normalize the traffic regarding the BACnet network layer, we first need to analyze the structure of the BACnet NPDU and the network layer addressing. Every BACnet device has a medium access control (MAC) address which is combined with the number of a BACnet (sub)net (the "network number") to form the network level address. The main task performed by BACnet routers is to forward packets to a single remote device and to broadcast packets to a remote network. Broadcasting is an essential feature in BACnet. Three types of broadcasts are supported: local, global, and remote broadcasting. *Local* broadcasts are received by all devices inside the local subnet. *Remote* broadcasts are received by all devices on a remote network and *global* broadcasts are received by all devices on all networks which are part of the BACnet internetwork.

In the remainder of the paper, we focus on BACnet/IP, i.e. BACnet encapsulated in UDP sent over IPv4, for which we define our normalization rules. Therefore, BACnet NPDUs are encapsulated into UDP (port 47808, or *0xbac0*). Fig. 2 visualizes the encapsulation of BACnet into the UDP payload [ND12a] and the structure of the BACnet NPDUs is shown in Fig. 3.

Figure 2: BACnet/IP in UDP datagram

The BACnet NPDU is split into two areas (Fig. 3). The first area is called the *network protocol control information* (NPCI) and the second area is called the *network service data unit* (NSDU). The NPCI contains the BACnet version, the NPCI control octet we describe separately, a hop counter, and address information (destination network (DNET), length of the destination address (DLEN), destination address (DADR), source network (SNET), length of the source address (SLEN), and source address (SADR)). The NSDU can either be a network layer message or it can be an APDU.

	Octet	Description
NPCI	1	Version
	1	NPCI Control Octet
	2	Destination Network (DNET)
	1	Dest. Address Length (DLEN)
	Variable	Destination Address (DADR)
	2	Source Network (SNET)
	1	Source Address Length (SLEN)
	Variable	Source Address (SADR)
	1	Hop Count
NSDU	Variable	Network Layer Message or Application Layer Protocol Data Unit (APDU)

Figure 3: BACnet NPDU format

The NPCI control octet contains 8 bits (0-7) and is shown in Fig. 4. Bit 7 acts as an indicator which conveys whether the NSDU contains a network layer message or an APDU. Bit 6 and 4 are reserved for future use. Bit 5 is used as a destination specifier to indicate whether DNET, DLEN, and DADR are present, or not. Bit 3 is used as source specifier to indicate whether SNET, SLEN, and SADR are present, or not. Bit 2 indicates whether a reply is expected, or not. Bit 1 and 0 are used to define priorities of BACnet messages like (11=*Life Safety*, 10=*Critical Equip*, 01=*Urgent*, 00=*Normal*).

Bit	7	6	5	4	3	2	1 0
Description	Indica-tion	Res.	Dest. Specifier	Res.	Source Specifier	Exp. Reply	Priority

Figure 4: BACnet NPCI control octet

If the destination for the message is a device within the same subnet, no additional network address is required. If the destination is within a remote network, the client device must include the destination network number and MAC address of the destination device. The router on the local network will insert the source network number automatically so that a response can be returned to the sender. Thus, a device does not need to know its own network number. The only addressing information required to send a message to a remote device is the remote device's network number and MAC address.

Cryptographic means in the BACnet network layer are optional. The general purpose is to provide *peer entity*, *data origin*, and *operator authentication* as well as *data confidentiality*, and *integrity mechanisms* in order to secure the message transfer at the network layer. These security mechanisms were deployed because network security standards like IPsec and Kerberos are not applicable for BACnet. BACnet security policies can define four security levels: *encrypted-trusted* (messages are encrypted and security on data link or physical layer is not required), *plain-trusted* (messages are sent in plain text but are physically secured, i.e. devices are protected from unauthorized physical access, *signed-trusted* (messages are signed and physical security is not required), and *plain-non-trusted* (messages are neither encrypted nor physically secured) [ND12b]. A security message is indicated as a network layer message in the NSDU. The different types of security messages are depicted in a so-called *Service Data* field which has got a variable length.

4 Attack Vectors, Vulnerabilities and Ambiguities related to BACnet

Looking at the structure of the protocol format, we can classify different areas which can be attack vectors, where vulnerabilities can be exploited, or can require unambiguity. The DIN EN ISO 16484-5 standard [ND12a] defines valid values for all fields in BACnet messages. Packets with a format that is incompliant to the standard are simply invalid and must not be accepted and should be dropped. In case a reply is anticipated by the sender and the packet got dropped by the normalizer, the normalizer can additionally return an error message. We want to rule out the following attack vectors for the network layer as well as an attack on BACnet's segmentation functionality:

1) Smurf Attack: An attacker can spoof the victim's IP address in order to misuse it. A classic example is a *smurf attack*: the attacker uses a victim's address as source address of several broadcast pings [Gu07]. The victim afterwards receives the answers for each request. If the number of messages sent to the victim exceeds its processing capabilities, the smurf attack can cause a denial of service. We adapt the smurf attack to BACnet. If an attacker is able to modify the the source (SADR and SNET) at the BACnet network layer, she will be able to spoof the address of broadcasted requests and can thus cause denial of service for selected BACnet devices.

2) Router Advertisement Flooding: A similar attack is possible if an attacker is able to spoof a target device's source address and source network (SADR and SNET) to send a *Who-is-Router-to-Network* message (requesting a router advertisement for a given network). The result is that the target will receive router advertisements from all the routers in the local network. If the attacker repeats this procedure and sends too many repeated messages, the target is likely to receive too many responses in a time window as it can handle, causing a denial of service.

3) Traffic Redirection: An attacker can spoof *Router-Available-to-Network* messages, i.e. messages indicating the availability of a router, with the goal to redirect selected traffic over itself in order to gain confidential monitoring data (e.g., presence sensor data of a given room to plan a physical break-in [Wen12b]).

4) Re-Routing DoS type 1: To cause a message flood on router *R*, an attacker can broadcast spoofed *I-Am-Router-To-Network* messages to a network using the source address of *R*. Therefore, all possible destination network addresses can be used as a parameter of the routing advertisement. This attack forces R to handle all responses for the *I-Am-Router-To-Network* message and moreover forces R to handle all remote traffic.

5) Re-Routing DoS type 2: If the target device of attack *4)* is *not* a router, the attacker can redirect all traffic for remote networks to a device incapable of forwarding messages to isolate the communication of a subnet.

We plan to limit the discussed attacks by defining appropriate limits for broadcast messages per time interval in Snort but cannot eliminate the attacks entirely.

It is important to mention that broadcast floods in BACnet networks can be caused by devices which are not configured properly. At least three examples from practice are

known: i) the wrong set-up of layer two switches that can lead to loops; ii) the use of multiple broadcasting devices (so-called *BACnet Broadcast Management Devices*, BBMDs) in a chain without a broadcast-limiting router device in the chain; or iii) the combination of BACnet/Ethernet ISO 8802-3 and BACnet/IP routers within the same infrastructure configured to utilize the same UDP port (leads to permanent broadcast exchanges between both layers).

6) Malformed Messages: Ensuring robustness for the protocol stacks of BACnet devices is essential as firmware is seldomly updated. Therefore, packets violating the rules of the BACnet standard must be modified or discarded to achieve unambiguity.

We moreover discovered an attack in the context of the application layer segmentation:

7) Inconsistent Retransmissions (BACnet Application Layer): We cover the generic problem of *inconsistent retransmissions* [Han01] in BACnet, i.e. (overlapping) segments sent twice to a destination. In IP, attackers are able to exploit vulnerabilities of packet fragmentation to gain additional privileges [Han01] and incorrect indication of associated fragments or incomplete fragments can additionally lead to a denial of service [Atl13]. To counter segmentation attacks in BACnet, we decided that segments containing invalid indication, i.e. those that comprise parts of already seen segments, are dropped. Precisely, the Segmentation Protocol Information in the APDU of a message must contain matching sequence numbers otherwise these messages are dropped.

5 Traffic Normalization for BACnet

We currently implement a Snort normalizer extension capable to normalize BACnet/IP traffic. At the moment of paper submission, we already support eight normalization rules. The Snort extension includes countermeasures for the discussed attack vectors and rules to remove ambiguities within the traffic. Our current testbed is implemented using the BACnet stack (cf. *bacnet.sourceforge.net*) and virtual machines which represent multiple BACnet devices each (Fig. 5). We use a self-written BACnet fuzzer to create incompliant BACnet messages, which are sent to the normalizer, and monitor the received messages at the destination host using *Wireshark*. As soon as our Snort normalizer is in a prototype state, we will extend our testbed with real BACnet hardware.

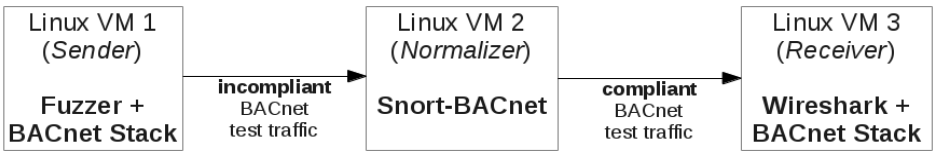


Figure 5: BACnet testbed used to develop the Snort normalizer extension

In the remainder, we introduce our normalization rules for the i*)* NPCI field, ii*)* for BACnet message types, *iii)* for BACnet security message types, and *iv)* for the handling of BACnet priority messages.

1) NPCI: Being an always present component of a BACnet NPDU, the NPCI field including the NPCI control octet must always be normalized.

Reasons to DROP messages:

1. Protocol Version Number != 0x01 (only BACnet version 1 is allowed)
2. DNET = 0, or SNET = 0, or SNET = 0xFFFF (forbidden by the ISO standard)
3. Multicasts and local broadcasts with DNET=0xffffffffffff using ISO 8802-3, DNET=0x00 using ARCNET, DNET=0xff using MS/TP, DNET=0x00 using LonTalk
4. Bit 3 of the NPCI control octet = 1 and SLEN = 0
5. Bit 7 of the NPCI Control Octet = 1 and a reply is expected (bit 2)
6. Deny PTP (Point-to-Point) connections if DNET = 0xFFFF

Reasons to MODIFY messages:

1. set DLEN = 0 and DADR=0 if the message is a remote broadcast
2. set bit 6 and 4 of the NPCI control octet to 0 as they are reserved for future use

2) BACnet Message Types: As explained in Section 3, the NPCI control octet (bit 7) indicates whether a BACnet message comprises a network layer message or an APDU. In general the number of network layer messages of the same type, so called *message types*, which are received, must be limited to counter broadcast floods. The watermark values depend on the particular BACnet devices and must thus be vendor-specific. In an empirical study, the MBS GmbH measured that most tested BACnet devices can process not more than 180 messages per second. We moreover determined the following normalization rules for the different network layer message types.

Rules for multiple message types:

1. Drop message if more than 3 bytes follow after message type and the message type is either I-Could-Be-Router-To-Network (0x02), Reject-Message-To-Network (0x03), Router-Available-To-Network (0x08), or Network-Number-Is (0x13).
2. Drop message if the total count of bytes is not even after message type and the message type is either I-am-Router-to-Network (0x01), Router-Busy-To-Network(0x04) or Router-Available-To-Network (0x05).
3. Drop message if more than (NUMBER_OF_PORTS x 4) + each PORT_INFO_-LENGTH bytes follow after message type and the message type is either Initialize-Routing-Table (0x06) or Initialize-Routing-Table-Ack (0x07).
4. Drop message if more than 2 bytes follow after message type and the message type is Who-is-Router-to-Network (0x00) or Disconnect-Connection-To-Network (0x09).

Rules for specific message types:

1. Who-is-Router-to-Network (0x00): Limit the number of messages to 180 per second (value is planned to be configurable to enable product-specific limits).
2. I-am-Router-to-Network (0x01): Drop message if not transmitted with a broadcast MAC address

3. What-Is-Network-Number (0x12): Drop message if not transmitted with a local broadcast or local unicast address [ND12a], or if Hop-Count > 0, or if SADR is the same during *n* minutes (*n* should be configurable)
4. Network-Number-Is (0x13): Drop message if SNET/SADR or DNET/DADR is set or if sent to local unicast address

3) Security Message Types: We moreover define rules for security messages as stated in [ND12b]. Error messages in each case shall be always sent *signed-trusted*.

Rules for multiple message types:

1. Drop message if broadcasted and message type is either Challenge-Request (0x0a), Update-Key-Set (0x0e), Update-Distribution-Key (0x0f), or Set-Master-Key (0x11).
2. Modify NPCI control octet: set 2nd bit to 1 if the message type is either Challenge-Request (0x0a), Request-Key-Update (0x0d), Update-Key-Set (0x0e), Update-Distribution-Key (0x0f), or Set-Master-Key (0x11).
3. Modify NPCI control octet: set 2nd bit to 0 if the message type is either Security-Response (0x0c) or Request-Master-Key (0x10).

Rules related to message size:

1. Drop message if more than 9 bytes follow after message type (Challenge-Request, 0x0a), if more payload is transferred than specified (Security-Payload, 0x0b), if more than 19 bytes follow after message type (Request-Key-Update, 0x0d), if more than 21 bytes + bytes of keys follow after message type (Update-Key-Set, 0x0e), or if more than 1 byte + bytes of key follow after message type (Update-Distribution-Key, 0x0f).

Rules for specific message types:

1. Security-Response (0x0c): Drop message if broadcasted or if (RESPONSE_CODE equals either 0x06 and RESPONSE_SPECIFIC_PARAMETERS > 4 bytes, or 0x07 and RESPONSE_SPECIFIC_PARAMETERS > 2 bytes, or 0x0f and RESPONSE_SPECIFIC_PARAMETERS > 2 bytes, or 0x15 and RESPONSE_ -SPECIFIC_PARAMETERS > 1 byte, or 0x16 and RESPONSE_SPECIFIC_- PARAMETERS > 3 bytes, or 0x17 and RESPONSE_SPECIFIC_PARAMETERS > 2 bytes, or 0x18 and RESPONSE_SPECIFIC_PARAMETERS > 1 byte, or 0x0e and RESPONSE_SPECIFIC_PARAMETERS not odd and first byte is not 0x0).

4) Priority Handling: BACnet allows assigning each message a priority [ND12a]. The last ascending two bits within the NPCI control octet indicate the urgency of a packet. The highest possible priority of a packet is the *life safety* message and can be handled in a prioritized way by receiving devices. However, in practice this feature is rarely used.

We aim on introducing normalization for packets of *all* priority levels if they are not well-formed according to the ISO standard [ND12a]. However, we therefore must take into account that *i)* not all malformed packets must be caused by an attacker and *ii)* that dropping a life safety message can result in significant side-effects if the message is not delivered but contains life-essential information.

Our intention is to affirm human safety with an additional function exemplified by the detection of toxic gas in a room. In case of an indoor gas detection, it is an urgent BACnet task to automatically close the door in the particular room to prevent expansion of the gas. It is evident that a general rule to drop malformed packets is not appropriate in such a case as dropping could prevent that the door will be closed and thus could affect the life of inhabitants. We state that life safety messages must be *modified* (in order to match compliant NPDUs according to the standard as closely as possible) and always forwarded, but should never be dropped. We are aware of the fact, that an attacker could explicitly take advantage of such a rule. Therefore, we propose to configure life safety messages with additional constraints. *First*, we propose to enforce that BACnet messages sent with life safety priority MUST always have a *trusted level*, i.e. they must be either encrypted or physically secured according to [ND12b], so that the receiving device is aware of the sending device. Life safety messages not encoded this way will be dropped. If the authentication is recognized, malformed packets are modified to a correct format. *Second*, we propose a reply mechanism. Messages sent with life safety priority should contain the expectancy for a reply. This feature could be activated for each message passing the normalizer by forcing a reply through the NPCI control octet. The normalizer would thus initiate the re-sending of unacknowledged packets as the original sender of the message is not aware of the normalized NPCI control octet.

6 Conclusion and Future Work

Building automation systems (BAS) are accessible over the Internet and by TCP/IP in-house networks. A problematic aspect is the vulnerability of BAS against typical TCP/IP-attacks such as spoofing and malformed packets (fuzzing). However, BAS control critical equipment (e.g., smoke-detectors) and applicability of protection means in practice is required to ensure reliable BAS.

We present vulnerabilities within the internationally standardized BACnet protocol used in many BAS worldwide and implemented by hundreds of vendors. These vulnerabilities comprise DoS attacks, spoofing attacks, routing attacks, and attacks based on malformed network messages. To counter these attacks, we propose to adapt the concept of traffic normalization from TCP/IP networks to BAS and analyze the network layer of BACnet in order to deduce several normalization rules. Our normalization rules additionally enforce compliant BACnet network messages in order to protect receiver devices with broken BACnet implementations against fuzzing attacks.

In currently ongoing work, we integrate BACnet support into the Snort normalizer. Besides, we plan to analyze various aspects of the BACnet application layer (especially safety and physically access control functionality) to determine further normalization rules. Moreover, we plan to actively introduce our work into the BACnet community to propose its partly integration into further versions of the BACnet standard.

References

[ASH13] ASHRAE: Proposed Addendum ai to Standard 135-2012, BACnet – A Data Communication Protocol for Building Automation and Control Networks. Atlanta, 2013.

[Atl13] Atlasis, A.: Fragmentation (Overlapping) Attacks One Year Later. Troopers 13 – IPv6 Security Summit 2013.

[Bac13] BACnet.org: BACnet Vendor IDs, http://www.bacnet.org/VendorID/BACnet%20Vendor%20IDs.htm, February 2013.

[Bow13] Bowers, B.: BACnet Attack Framework, slides of a talk at Shmoocon, 2013.

[GKNP06] Granzer, W., Kastner, W., Neugschwandtner, G. and Praus, F.: Security in networked building automation systems. In Proc. 2006 IEEE International Workshop on Factory Communication Systems, pp. 283–292, 2006.

[Han01] Handley, M., Paxson, V. and Kreibich, C.: Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics. Berkeley, 2001.

[HBG06] Holmberg, D. G., Bender, J., and Galler, M.: Using the BACnet firewall router. BACnet Today (A Supplement to ASHRAE Journal), pp. B10–B14, November 2006.

[Hol03] Holmberg. D. G.: Enemies at the gates. BACnet Today (A Supplement to ASHRAE Journal), pp. B24–B28, 2003.

[MW+00] Malan, G.R., Watson, D., Jahanian F. and Howell, P.: Transport and application protocol scrubbing, *Proc. of the INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 1381-1390,* 2000.

[ND12a] Normenausschuss Heiz- und Raumlufttechnik (NHRS), Deutsche Komission Elektrotechnik Elektronik Informationstechnik (DKE): Systeme der Gebäudeautomation – Teil 5: Datenkommunikationsprotokoll (ISO 16484-5:2012); pp. 66-88, The Network Layer, 2012.

[ND12b] Normenausschuss Heiz- und Raumlufttechnik (NHRS), Deutsche Komission Elektrotechnik Elektronik Informationstechnik (DKE): Systeme der Gebäudeautomation – Teil 5: Datenkommunikationsprotokoll (ISO 16484-5:2012); pp. 649-700, Network Security, 2012.

[Gu07] Gu, Q. and Liu, P.: Denial of Service Attacks, Texas University, http://s2.ist.psu.edu/paper/DDoS-Chap-Gu-June-07.pdf, 2007.

[SZ12] Soucek, S. and Zucker, G.: Current developments and challenges in building automation. e & i (Elektrotechnik und Informationstechnik), 129(4), pp. 278-285, 2012.

[Tom12] Tom, S.: BACnet for a City – Saving Energy one Small Building at a Time; BACnet Today and the Smart Grid | A Supplement to ASHRAE Journal, November 2012.

[Wen12a] Wendzel, S.: The Problem of Traffic Normalization in a Covert Channel's Network Environment Learning Phase, in Proc. *Sicherheit 2012*, pp. 149-161, LNI 195, GI, 2012.

[Wen12b] Wendzel, S.: Covert and Side Channels in Buildings and the Prototype of a Building-aware Active Warden, in Proc. *ICC Workshop on Security and Forensics in Communication Systems (SFCS'12)*, Ottawa, pp. 6753-6758, IEEE, 2012.

[WKR12] Wendzel, S., Kahler, B., Rist, T.: Covert Channels and their Prevention in Building Automation Protocols – A Prototype Exemplified Using BACnet, in Proc. *CPSCom Workshop on Security of Systems and Software Resiliency (3SL'12)*, Besançon, IEEE, pp. 731-736, 2012.

An efficient approach to tolerate attackers in fault-tolerant systems

Johannes Formann
Universität Duisburg-Essen, Essen
formann@dc.uni-due.de

Abstract: Malicious attackers can cause severe damage (financially or to the environment) if they gain control of safety-relevant systems. This paper shows why the traditional disjoint treatment of security and fault tolerance has weaknesses if the attacker gains access to the fault tolerant system and how an integrated approach that utilize existing fault tolerance techniques could be an effective security mechanism. An efficient integrated safety and security approach is presented for fault tolerant systems, which achieves protection against attacks via the network by forming a logically isolated (sub-) network which is resilient against a bug in the codebase. Isolation is obtained by diverse design of a general reusable (software and/or hardware) component that prevents any unauthorized message transfer towards the secured application program. Messages from other compromised nodes are tolerated utilizing existing majority voting mechanism.

1 Introduction

Modern distributed safety-relevant systems utilize different techniques to ensure fault tolerance. They usually assume that their communication network is sufficiently isolated, and thus protected against launching of malicious attacks. However in various domains the communication systems are more and more opened to external networks. In industrial automation the so-called horizontal and vertical integration opens the network to other automation systems as well as to administration and office networks. Moreover, studies show [BE04, SM07] that these networks are often not very secure.
In some domains, e. g. automotive or energy distribution, modern networks are designed to have communication with the outside world (e. g. Car-to-Car or Car-to-Infrastructure [HKD08, SR10]). Some voices say that the increasing connectivity could enable attacks even in safety-relevant systems, while others still assume the network to be kept isolated through out its entire lifespan.
The naive approach of protecting the network by a firewall has different drawbacks. One of the main drawbacks lies in the fact that the complete communication has to pass the firewall. However, as described in [BE04], there are often many communication channels that must be secured. Another drawback: the firewall must be able to distinguish between valid and malicious traffic. To ensure this in the case an attacker has gained control of a legitimate sender or in the case of masquerade the firewall needs to implement checks for the payload. This means a firewall has to be designed individually for the specific applications. This results in high costs (no “component of the shelf”) and increases the

probability that the firewall itself might have a bug that enables an attacker to access the internal network. In this paper we assume that it is possible for an attacker to gain access to the network of a safety-relevant system, since there are many examples (see appendix A.1) that proves this is a valid assumption.
We will show an approach to logically isolate the safety-relevant system and make it resilient against an attacker which has access to a remotely exploitable vulnerability, even if the vulnerability is in the code handling the security mechanism. The approach is based on diversely developed communication hypervisors and majority voting. The solution is completely implemented in an application-independent component. Thus it is reusable for any application.

2 Why redundant replicas do not solve the problem

At a first glance an attacker with knowledge of just a single exploitable vulnerability in the software is not a fundamentally different threat to a system compared to other threats from the environment (radiation, thermal stress, ...). Both can result in erroneous internal states [Jon99] of the program, which in turn can result in wrong behavior (e.g. malicious byzantine faults). These effects are regularly considered in fault-tolerant systems [Ech90]. However an attacker can be even worse since his behavior might not match the assumptions typically made for the tolerance of technical faults. Examples are addressed in the following sections.

2.1 Number of affected nodes

In most systems redundancy for fault tolerance is accomplished by using n replicas of a node. Usually n is $2f + 1$ (in case of majority voting) or $3f + 1$ (in case of Byzantine agreement) assuming up to f faulty nodes. A bug is a systematic fault [KA83, CMSB08, KL86] that has been introduced while programming or setting up the system. Normally the program is replicated for all redundant nodes. Consequently, nothing prevents the attacker to take control of all n replicas. This clearly violates the assumption of a maximum of f faulty nodes. Since the attacker can control the majority (or all) nodes, he can control the complete system. In [GÖ3] an approach has been proposed to model the security and safety over time in the presence of an attacker.
An exception of this rule is the case where redundant nodes own independently/diversely developed program versions rather than instances of the very same program. When the programs in the redundant nodes differ from each other, a common design fault is still possible, but not likely to occur [KA83, KL86, ALS88].

2.2 Assumptions regarding the behavior

In many systems there are not only limitations regarding the number of tolerated faults, there are also assumptions with respect to the behavior a faulty node can exhibit. A fre-

quently used assumption says that faults of nodes are stochastically independent [Sch90]. If the behavior is "worse" than the assumed one, fault tolerance is not guaranteed. If these assumptions do not include malicious Byzantine faults (if technically possible) and the possibility that all faulty nodes cooperate, the assumptions are likely to be violated by a malicious attacker who controls f faulty nodes.

2.3 Resilient communication system

A distributed fault-tolerant system usually depends on its communication infrastructure. Many systems are designed to tolerate a loss of one communication channel (e.g. FlexRay [Fle05] configured to use both buses for redundancy) but usually the fault model assumes one or more random errors in the communication system, not a malicious attacker. Most communication systems have some kind of communication controllers (FlexRay, CAN [Bos91]) which are usually configured by the node they are attached to. In this case a malicious attacker can simply reconfigure its communication controller to have the biggest possible impact on the communication system. Very simple communication systems like LIN [LC10] can be (almost completely) implemented in software. Such communication systems are designated for disturbing the communication by an attacker. It has been shown for different bus systems, that effective denial of service attacks are possible [WWW07, JM06] by capturing only a single node.

3 Definitions

3.1 Program

A program provides a function in a system and has clearly defined interfaces to the outside world. In this Paper the interface is assumed to consist only of message transmission over the communication system (to exchange information with other programs in different nodes) and of interaction with the outside world by simple[1] IO-Interfaces. Since simple IO does not provide any means to transport a specially crafted payload that can be used to exploit vulnerabilities in the program and thereby modify the behavior, they do not need any further protection.
A program can be designed to work in a fault-tolerant environment e. g. by using multiple program instances (distributed over redundant nodes), majority voting for the input, and multicasting of results to multiple instances of the program that further processes the data (also distributed over redundant nodes).
A program is assumed to be fault-free, if the majority of the program replicas work correctly, so that the output can be sent to voters, which will mask out wrong results. Typically the voters are located at the input side of the replicas of the program that processes the output data further.

[1] No communication protocol is used on top.

3.2 Node

In this paper a node is a physical entity that is at least composed of a microcontroller and an interface to the communication system. A node might only support a single specialized application (e. g. a node that is built to read a value from a sensor and then transmit the value to other nodes) or utilize some kind of operating system to support different local applications (e. g. a so-called PLC (programmable logic controller. In this work a single node is also the smallest entity an attacker could gain access to. It is assumed that access is achieved by exploiting a bug regardless if the bug is in the application or the operating system (if present). In any case we assume the attacker gains full control of the node.

3.3 System

A system delivers at least one (fault-tolerant) service to the outside world. To provide a service the system can be internally subdivided into programs. In this paper it is assumed that a system is working properly, if all programs are working correctly.
To ensure the logical isolation it is assumed, that the set of nodes that runs the programs are not used for other systems. Furthermore, it is assumed that each node could have one or more potentially exploitable vulnerabilities. In all, the system consists of different (potentially intersecting) subsets of nodes owning the same exploitable bug, which can be misused over the network to obtain control over the node. To avoid making assumptions regarding the vulnerability, we assume – as said before – that the attacker can take full control of a node if he has knowledge how to exploit a vulnerability.

3.4 Attacker

The attacker is an entity trying to modify the system behavior to operate outside the designed safe envelope. It can utilizes intelligence and external resources (e.g. computing power) to archive this goal.

3.5 Communication system

The communication system ensures that a message is delivered from the sender to the receiver. The necessary properties depend on the service that the system has to deliver. If the service has no safe state that can be reached easily, it is usually called a “fail-operational” service. That means the system must deliver the service even in the case of an error. In this case communication has to be resilient against denial of service attacks. However, if the service can reach a safe state, the system is usually designed with watchdogs, which ensure the safe state is activated in the case of a communication breakdown. In this case no protection against denial of service is required.
The attacks on the communication system itself and the respective countermeasures are not in the focus of this paper.

4 Cryptographic Island

In this chapter the new approach is presented, that leverages existing fault tolerance techniques to create an application independent framework which provides protection against an attack over an existing communication system.
The cryptographic island utilizes two basic design principles. Logical isolation through cryptographic signatures isolates the application from external threats. To avoid a common vulnerability in the code that ensures the logical isolation, a diversely developed framework/hypervisor is utilized to enforce the logical isolation. To reduce the necessary development effort the framework/hypervisor should be designed in such a way, that it has a generic interface towards the programs, so it can easily be reused.

4.1 Assumptions

The concept of the cryptographic island relies on some assumptions.

- The attacker may somehow gain access to the communication system, but the he/she has no physical access to any of the nodes of the safety-relevant system. For example the attacker might have gained control of another node (e. q. some monitoring service used for maintenance) connected to the communication system or found an unguarded connection to this network.
- The communication system provides the needed level of robustness against the attacker (see chapter 3.5).
- There is no common bug in the diverse framework variant. According to [KA83, KL86, ALS88, GV79] common bugs are very unlikely. To reduce the probability that an (unlikely) common bug can be utilized by the same exploit method to gain control of the node, different "tool chains" (i. e. programming languages, libraries, compilers) could be used. In this case there is a good chance that an exploit works only for one framework variant, and the other variants are not affected (message is detected as being erroneous) or they expose a much easier to handle fail-silent behavior and simply crash.
- There are no configuration errors. This means especially that all replicas of a process are distributed on nodes with different framework variants. Automated checks could be developed to ensure this property at system startup.
- The attacker has only knowledge of a known and limited number of vulnerabilities that he/she can use for gaining access. If there are more exploitable bugs, the other vulnerabilities are unknown to the attacker. This is the equivalent to the f fault assumption for fault tolerant systems.
  Without making further assumptions regarding the application behavior running on top of the framework, the number of vulnerabilitie sis limited to one. This assumption could be considered valid if further techniques like those presented in [SPS08, CS11] are used to detect the presence of an attacker on the compromised

node. After detection one can either direct the system into a safe state within a time period that is too short for an attacker to find a second exploitable vulnerability. Alternatively one can reconfigure the system by using spares (this means: additional diverse variants of the framework) or repairing the system in a timely matter by fixing the bug so that all assumptions are met again and the compromised nodes are no longer part of the system.
Different from the generic case, two approaches to tolerate more than one known bug are presented in chapter 4.4.

4.2 Logical isolation

As stated in the introduction, an isolated environment is a good protection against malicious attackers. Since physical isolation is not always feasible, logical isolation can be utilized. Logical isolation can be achieved by the combination of an effective protection against masquerading, and list of legitimate senders for each program instance.
Then for all incoming messages a node knows the real senders and it is able to decide whether it wants to accept a message or to drop it.
An effective protection against masquerading could be implemented in the communication system. If the communication system does not offer an effective masquerade protection (and most communication systems do not) the masquerade protection must be implemented inside the framework. Strong cryptographic signatures are an effective method to verify the identity of the sender from a message. Some approaches have already been discussed for embedded (realtime) networks [GR97, SK09, EKP+07, WWP04]. The strength (usually measured in key-length) of the used cryptographic methods depends on the computing power of the attacker. Typically the computing power of the attacker has to be assumed unknown, and consequently considered to be large.

4.3 Reusable diverse framework

To protect the system, it is necessary that the application is protected against unauthorized senders, that may know a bug in the application. But the attacker might also know a bug inside the logic that checks the cryptographic signatures or the message handling before the signature check. To ensure that the system is resilient against both kind of attacks, the protection of the application can be implemented using a framework that acts as a kind of hypervisor checking every communication. To mitigate bugs in the hypervisor, it is recommended that different variants of this framework are developed using diverse programming. Thus a common vulnerability in two or more different variants of the framework is very unlikely. To create a fault-tolerant application that is also resilient against known bugs the different replicas of the same program have to be executed on top of different diverse variants of the framework.
Figure 3 in appendix A.2 show the possible locations of the framework within a node. The framework can either be purely implemented in software (see figure 3a) or comprise both software and hardware (see figure 3b).

For an example assume that programs P and Q have to be executed by three redundant replicas each. Moreover, assume that diverse variants A, B and C of the framework have been implemented. Then the replicas P_1, P_2 and P_3 of P, and Q_1, Q_2 and Q_3 of Q may be executed as follows: replicas P_1 and Q_3 on top of A, replicas P_2 and Q_2 on top of B, and replicas P_3 and Q_1 on top of C. Permutations within (P_1, P_2, P_3) or within (Q_1 Q_2, Q_3) will also work, of course. The allocation to nodes is not determined by this approach. It can be chosen according to performance considerations or to the available IO connections. Let N_1, N_2, N_3, N_4 and N_5 be a set of five nodes. Then one of many potential allocations is the following (see figure 1):

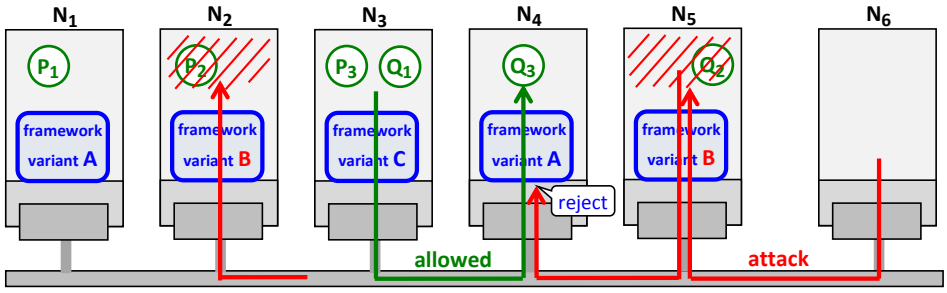


Figure 1: Diverse framework in the example system

nodeN_1 runs A and P_1 on top of it
node N_2 runs B and P_2 on top of it
node N_3 runs C and P_3 and Q_1 on top of it
node N_4 runs A and Q_3 on top of it
node N_5 runs B and Q_2 on top of it

Any node does not run more than one variant of the framework.

If an attacker has found an exploitable vulnerability in one of the framework variants (variant B in the example depicted in figure 1), only the replicas on top of this single framework variant are affected (which means that all nodes running this particular variant can be compromised). If the degree of redundancy is sufficient, then for each process only the minority of variants is affected. Thus majority voting will decide for the correct result despite the attack. In figure 1 the majority P_1 and P_3 of process P (using variants A and C respectively) and the majority Q_1 and Q_3 of process Q exist and produce correct results despite the attack.

To ensure this isolation, messages are signed using strong cryptography by the sender. Each receiver can check the true origin of a message it obtains and, moreover, can check if the message has not been modified during transmission via the network. The framework on the receiving nodes checks for each message if the stated sender is an allowed sender and if the signature of the message is valid for the stated sender. If both checks are passed successfully the message is delivered to the program. Otherwise it is discarded. It is assumed that the information required for these checks (the necessary cryptographic certificates of the valid senders) have been distributed safely, for example by deploying during the initial system deployment. The limitation to trusted and well-known sender result in logical point-to-point connections for the programs.

If a system consists of programs each of which is made fault-tolerant by replication, and,

moreover, the replicas of a program are distributed among the nodes in such a way, that each replica utilizes a different version of the framework, then the system is safe against an attacker that has knowledge of how to exploit a single vulnerability. The attacker is unable to control the majority of the replicas.
Assuming the key distribution is done without carelessness, an attacker can't attack the application itself, since the message is dropped due to the lack of a valid signature. Therefore the only chance to influence the system is to find a bug in one of the diverse framework variants that guards the message flow. If the attacker gains access to one remotely exploitable vulnerability, he/she could take control over all nodes running the same framework variant. If the system is designed as described, this will only affect a minority of the replicas of each application. So the potential erroneous results created by the attacker-controlled nodes could be easily tolerated in the applications like any other erroneous result (e. g. by applying majority voting).
Since only a single remotely exploitable bug is assumed, the attack is unsuccessful.
The development of a framework that acts as a hypervisor for message transmissions has two main advantages over complete diverse development. The first advantage is that once this framework has been developed, it can be used for different programs and systems. This reuse significantly reduces the costs for each project and potentially the framework versions could become "components of the shelf" if the environment is standardized like the AUTOSAR [AUT10] environment for the automotive domain. The second advantage is that only a small part of the functionality must be developed in a diverse way. This functionality is not necessarily implemented in software. Some parts of the functionality (e. g. the computation of the cryptographic signature due to performance reasons) or even the full functionality can be implemented in hardware.
The application programs themselves need not be programmed diversely. This further reduces the necessary effort.

4.4 Tolerating multiple bugs

Multiple bugs can be tolerated if the framework ensures, that the attacker is unable to send an exploit to a replica of the application on a unexploited framework version. This results in two requirements:

- The attacker must be unable to send a message the the protected application on nodes with an unaffected framework version while there are still known bugs.
- The attacker must be unable to determine a input signal that is required for the application. This usually means that the attacker is not allowed to control the majority of the nodes sending the input signal.

The basic approach does not ensure this property. An attacker with knowledge of two (or more) exploitable bugs might use the first exploit to gain access to a node that sends messages to all replicas of an application. If the second exploit works against the application, the attacker gains access to all replicas of this application, since he is able to sign the messages with the legitimate key he gained by accessing the first attacked node. With access

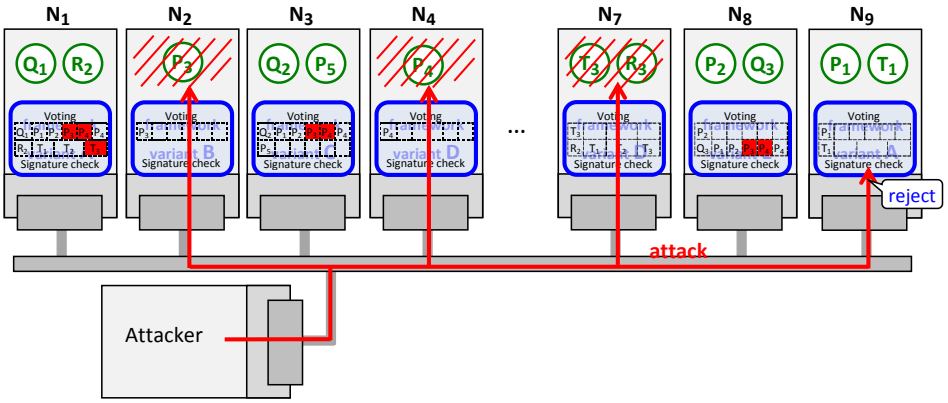


Figure 2: Example for voting inside the framework

to all replicas of a program, he could determine the output of all the replicas and therefore, independent of the number of tolerated faults, influence the system behavior.
Currently two different approaches for tolerating multiple bugs are research topics. The first approach is voting inside the framework and the second approach is usage of different framework variants on top of each other to create multiple protection layers. Possible realizations of the multi-layered approach are subject of ongoing research.

4.4.1 Voting inside the framework

To ensure, that the protected application does not receive an erroneous message from the attacker, the voting algorithm used by the fault-tolerant application can be placed inside the framework. When a message arrives the signature is checked. If the signature is valid the message is stored inside the framework. After the messages of all replica of a input program have arrived (or a timeout has expired) the framework executes a voting algorithm and delivers the result to the application. Messages from nodes controlled by the attacker are treated like all other (faulty) input messages, as long as the faulty/malicious inputs are the minority, they are discarded.
If there are $2f + 1$ replicas of the protected application on $2f + 1$ framework variants (with f being the number of faults including the exploitable bugs known to the attacker) the attacker could not attack the application, since he cannot affect the majority of used framework variants. If the used input messages are protected the same way ($2f+1$ replicas of the application generating the input on $2f + 1$ framework variants) the attacker could not control the majority too. Consequently the attacker can only control a minority of input messages, that are being discarded during the majority voting inside the fault-free framework versions.
In figure 2 an example is shown which shows an excerpt from a larger system with four applications (P, Q, R, T). The replicas of application P generate the input for the replicas of application Q and the replicas of application T generate the input for the replicas of application R. The inputs for P and T are not shown in this figure and the corresponding

fields in the tables are left blank. The applications P and Q tolerate two bugs or faults (resulting in five replicas running on top of five framework versions) and the applications R and T tolerate one bug/fault (resulting in three replicas running on top of three different framework versions).
The voting have to be programmed diversely as the rest of the framework, to reduce the risk that an attacker may find a common bug inside the majority voter. To enable reusability and keep the message handling logic small the voting algorithm inside the framework should be sufficiently simple (e.g. majority voting).
This approach has the advantage, that it is possible to tolerate more than one known bug, but it sets a few constraints for the applications. The application have to be designed to delegate voting to the framework, therefore the framework can not designed as a "drop-in-replacement" for existing libraries without protection. The application is restricted to simple (majority) voting algorithm. Sophisticated algorithms based on system behavior models would contradict the reusability and increase the codebase which increases the likelihood of bugs.

4.4.2 Multiple protection layer

Multiple protection layers stacked on top of each other have basically two advantages compared to the voting inside the framework:

- There are no assumptions regarding the program behavior. The program can use as complex voting algorithms as desired and the framework stack could be designed to be a "drop-in" replacement for the existing message sending and receiving API.
- The number of tolerated bugs could differ from the number of tolerated faults.

The main problem with this approach is caused by the fact, that the layers have to be independent from each other. If the attacker finds a bug in a protection layer, he/she must not be able to manipulate the other protection layers. This clearly violates the assumption in chapter 3.2 that an attacker has access to all software running on a node, once he gained access. There is ongoing research for system designs, where a different assumption for the access after an exploit could be justified.

5 Conclusion

Malicious attackers are a real world problem when designing modern safety-relevant systems. It attracts more and more attention of designers who are confronted with both safety and security issues. If the classical approach of full diversely developed software is not feasible, the new approach of forming cryptographic island provides a solution for logical isolation of the application programs against external attackers. The approach achieves a good tradeoff between security and necessary development effort leveraging the existing mechanism for fault tolerance and requires only a small reusable framework.
To make this approach usable for a broader set of use cases a special focus is set on the admission of external input from (potentially) untrusted systems via the communication network. This is a necessity for all scenarios where an external input (e. g. production

planning systems of a chemical plant, infrastructure-to-car communication in the automotive application) must be able to modify the behavior of the system in a "safe envelope" without affecting the safety and security goals.

References

[ALS88] A Avizienis, MR Lyu, and W Schutz. In search of effective diversity: a six-language study of fault-tolerant flight control software. In *The Eighteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, volume 1, pages 15–22. IEEE Comput. Soc. Press, 1988.

[AUT10] AUTOSAR. Specification of SW-C End-to-End Communication Protection Library, October 2010.

[BE04] Eric Byres and P Eng. The Myths and Facts behind Cyber Security Risks for Industrial Control Systems The BCIT Industrial Security Incident Database (ISID). In *VDE Kongress*, pages 1–6, 2004.

[Bos91] R Bosch. CAN specification version 2.0, September 1991.

[BS10] WILLIAM J. BROAD and DAVID E. SANGER. Worm Was Perfect for Sabotaging Centrifuges, November 2010.

[CMSB08] Byung-gon Chun, Petros Maniatis, Scott Shenker, and U C Berkeley. Diverse Replication for Single-Machine Byzantine-Fault Tolerance Challenges in a BFT Server. In *ATC'08 USENIX 2008 Annual Technical Conference on Annual Technical Conference*, pages 287–292, Berkeley, CA, 2008. USENIX Association.

[CS11] Ang Cui and S Stolfo. Defending embedded systems with software symbiotes. In Robin Sommer, Davide Balzarotti, and Gregor Maier, editors, *Recent Advances in Intrusion Detection*, pages 358–377, Menlo Park, CA, USA, September 2011. Springer Berlin Heidelberg.

[Ech90] Klaus Echtle. *Fehlertoleranzverfahren*. Studienreihe Informatik. Springer, 1990.

[EKP+07] Thomas Eisenbarth, Sandeep Kumar, Christof Paar, Axel Poschmann, and Leif Uhsadel. A Survey of Lightweight-Cryptography Implementations. *IEEE Design & Test of Computers*, 24(6):522–533, November 2007.

[Fil11] Jonathan Fildes. Stuxnet virus targets and spread revealed, November 2011.

[Fle05] FlexRay Consortium. FlexRay Communications System - Protocol Specification Version 2.1, December 2005.

[FMC11] Nicolas Falliere, LO Murchu, and Eric Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security ...*, 4(February):1–69, 2011.

[GÖ3] FC Gärtner. Byzantine Failures and Security: Arbitrary is not (always) Random. In Rüdiger Grimm, Hubert B. Keller, and Kai Rannenberg, editors, *GI Jahrestagung (Schwerpunkt "Sicherheit - Schutz und Zuverlässigkeit")*, pages 127–138. Swiss Federal Institute of Technology (EPFL), School of Computer and Communication Sciences, GI, 2003.

[GR97] Rosario Gennaro and Pankaj Rohatgi. How to sign digital streams. In *Advances in Cryptology - CRYPTO'97*, pages pp 180–197, Santa Barbara, California, USA, 1997. Springer Berlin Heidelberg.

[GV79] L Gmeiner and U Voges. Software diversity in reactor protection systems: An experiment. In R. Lauber, editor, *Safety of Computer Control Systems (Proceedings of IFAC Workshop on safety of computer control systems)*, pages 75–79. Pergamon Press, 1979.

[HKD08] Tobias Hoppe, Stefan Kiltz, and Jana Dittmann. Security threats to automotive CAN networks - practical examples and selected short-term countermeasures. In *Computer Safety, Reliability, and Security*, pages 235–248. Springer Verlag Berlin-Heidelberg, 2008.

[IC12] ICS-CERT. KEY MANAGEMENT ERRORS IN RUGGEDCOM'S RUGGED OPERATING SYSTEM, August 2012.

[Jc 12] Jc (JC CREW). Undocumented Backdoor Access to RuggedCom Devices, April 2012.

[JM06] Michael Jenkins and SM Mahmud. Security needs for the future intelligent vehicles. In *2006 SAE World Congress*, number 724, Detroit, MI, USA, 2006.

[Jon99] Erland Jonsson. On the functional relation between security and dependability impairments. *1999 workshop on New security*, pages 104–111, 1999.

[KA83] JPJ Kelly and A Avizienis. A specification-oriented multi-version software experiment. *Digest of Papers FTCS-13: Thirteenth International ...*, pages 120–126, 1983.

[KCR+10] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. Experimental Security Analysis of a Modern Automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE, 2010.

[KL86] John C. Knight and Nancy G. Leveson. An experimental evaluation of the assumption of independence in multiversion programming. *IEEE Transactions on Software Engineering*, SE-12(1):96–109, January 1986.

[Lan11] Ralph Langner. Stuxnet: Dissecting a Cyberwarfare Weapon. *IEEE Security & Privacy Magazine*, 9(3):49–51, May 2011.

[LC10] LIN-Consortium. LIN specification package (Rev 2.2A), 2010.

[Sch90] F.B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys (CSUR)*, 22(4):299–319, 1990.

[SK09] Christopher Szilagyi and Philip Koopman. Flexible multicast authentication for time-triggered embedded control network applications. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 165–174. IEEE, June 2009.

[SM07] Jill Slay and Michael Miller. Lessons learned from the maroochy water breach. *Critical Infrastructure Protection*, 253:73–82, 2007.

[SPS08] A Shah, Adrian Perrig, and Bruno Sinopoli. Mechanisms to provide integrity in SCADA and PCS devices *. In *International Workshop on Cyber-Physical Systems - Challenges and Applications (CPS-CA '08)*, Santorini, Greece, June 2008.

[SR10] Hendrik Schweppe and Yves Roudier. Security issues in vehicular systems: threats, emerging solutions and standards. *Presented at SAR-SSI*, pages 1–5, 2010.

[Tes13] Hugo (n.runs AG) Teso. Aircraft Hacking - Practical Aero Series, April 2013.

[WWP04] Marko Wolf, André Weimerskirch, and Christof Paar. Security in automotive bus systems. In *Proceedings of the Workshop on Embedded Security in Cars (escar)'04*, pages 1–13, 2004.

[WWW07] Marko Wolf, André Weimerskirch, and Thomas Wollinger. State of the Art: Embedding Security in Vehicles. *EURASIP Journal on Embedded Systems*, 2007:1–16, 2007.

[ZK13] Zeljka Zorz and Berislav Kucan. Hijacking airplanes with an Android phone, April 2013.

A Appendix

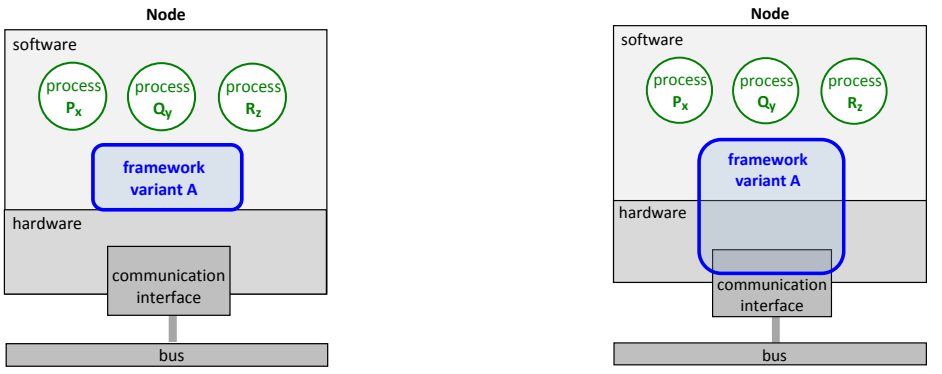
A.1 Examples of attacks to safety relevant systems

The following examples show, that the risk of unauthorized access is real and should be dealt with when designing a system.

- The StuxNet attack [FMC11, Lan11] on the uranium enrichment plant in Iran is well-known and got even attention in the mainstream press [Fil11, BS10]. It utilizes different exploits to gain access to the controllers and to manipulate them without triggering the emergency shutdown system.
  This attack demonstrates that even highly secured systems that have probably physically isolated networks are vulnerable against attacks.

- A recent talk [Tes13, ZK13] at the Hack In The Box Conference in Amsterdam has demonstrated a successful attack on a (specific model of an) autopilot for a plane from the ground utilizing the wireless communication systems ADS-B and ACARS that are installed in nearly all commercial planes and are active during the flight. The attack enabled modification of the plane's course and the flight level (the altitude). This demonstrates that even in industries that obey very high safety standards security problems may occur.
- An investigation in the year 2010 of a 2009 car model [KCR+10] showed that even without the presence of Car2Car or mobile internet access an attacker could manipulate the car (including acceleration, breaking, disabling the brakes) by exploiting different vulnerabilities.
  It has been shown that the starting point of the attack could be any device connected to the internal networks. In this study for most of the experiments a small WiFi enabled OBD-Plug in the engine compartment has been used. There was also a proof of concept, where the radio, which is connected to internal bus systems to display the current song in the dashboard, was the entry point. The attack was possible since the radio had a vulnerability that could be exploited with a modified audio CD.
- A former contractor used the wireless network in a water treatment plant to gain access. Over several weeks he pumped thousands cubic meters of untreated wastewater into a river [SM07].
- In the operating system of Programmable Logic Controllers [IC12, Jc 12] remotely exploitable bugs have been found, that are independent from the used application.

A.2 Illustration



(a) Implemented in software

(b) Implemented in hard- and software

Figure 3: Framework impkementations

Security Workflow Analysis Toolkit

Rafael Accorsi, Julius Holderer, Thomas Stocker, Richard Zahoransky

Abteilung für Telematik, Albert-Ludwigs-Universität, Freiburg
{accorsi,holderer,stocker,zahoransky}@iig.uni-freiburg.de

Abstract: Dieser Beitrag stellt das „Security Workflow Analysis Toolkit“ (SWAT) vor, eine Plattform für die formal fundierte Analyse von Geschäftsprozessen. Ausgehend von als Muster formalisierten Sicherheitsanforderungen dient SWAT als Basis für die Analyse von Prozessmodellen und Prozesslogs. Der vorliegende Beitrag zeigt anhand von Beispielen, welche Arten von Analysen mit SWAT möglich sind und wie SWAT hinsichtlich seiner Architektur aufgebaut ist.

1 Einleitung

Über 70% der Geschäftsprozesse werden in IT-gestützten Managementsystemen (BPMS) abgebildet, zumindest teilweise automatisiert ausgeführt und verwaltet [WH10]. Diese Umsetzung ermöglicht die flexible, adäquate Anpassung von Prozessen an geschäftliche Veränderungen und die bedarfsgerechte Einbindung externer Ressourcen. Allerdings stehen den geschäftlichen Chancen des Prozessmanagements erhebliche Risiken bzgl. der Einhaltung von Sicherheitsrichtlinien und gesetzlichen Vorschriften gegenüber. Wollen Unternehmen bspw. ihre Prozesse vollautomatisiert ausführen oder in die Cloud auslagern, müssen sie über Garantien verfügen, dass entsprechende Vorgaben nicht verletzt werden bzw. dass Mechanismen vorhanden sind, die solche Verletzungen zeitnah detektieren [CGJ+09].

Dies betrifft insbesondere den Aspekt der Kontrolle und Audit der Sicherheit und der Privatsphäre [Acc13, MAHS10]: es muss gewährleistet sein, dass verarbeitete Informationen – wie Kundendaten, finanzielle Transaktionen und geschäftliche Interna – geschützt bleiben und die eigenen Prozesse vor fremdem Einfluss oder internen Angreifern abgeschottet sind. Das Vertrauen in bestehende Lösungen ist bislang gering [MA13]. Mangelnde Sicherheits- und Compliance-Garantien werden derzeit in der Industrie als großes Hemmnis für den betrieblichen Einsatz automatisierter Prozesse angesehen [Sta11, LA11].

Dieser Artikel stellt das „Security Workflow Analysis Toolkit“ (SWAT) vor, eine Plattform für die formal fundierte Analyse von Geschäftsprozessen. SWAT verfügt über Verfahren, die sowohl Prozessmodelle als auch zur Laufzeit generierte Ereignislogs analysieren können. Dabei können die Prozessausprägungen auf die Einhaltung nicht-funktionaler, als Muster formal erfasster Eigenschaften geprüft werden. Konkret bietet SWAT folgende Besonderheiten:

- Die Formalisierung von Eigenschaften anhand vordefinierter Muster erleichtert die Spezifikation von Anforderungen: die Muster müssen für die Analyse lediglich mit kontextspezifschen Parametern instanziiert werden. Die vorhandenen Muster erfassen vollständig die strukturellen Kontrollflussbedingungen nach [DAC98], ebenso wie Compliance-[ALS11] und Sicherheitsanforderungen [AW11].
- Die Analyse baut derzeit auf erprobten Mechanismen für die Verifikation von (gefärbten) Petrinetzen, wie LoLA[1] und PRISM[2], oder Prozesslogs, für welche SCIFF[3] verwendet wird, auf.
- Die Analyseergebnisse werden visualisiert und somit für den Benutzer verständlich gemacht. Gegenstand der Visualisierung sind Evidenzen, welche auf die Verletzung von Anforderungen hinweisen.

Artikelaufbau: Abschnitt 2 gibt einen Überblick der in SWAT vorhandenen Funktionalitäten. Abschnitt 3 zeigt anhand von Beispielen auf, welche Analysemöglichkeiten derzeit bestehen und Abschnitt 4 beschreibt die SWAT-Architektur. Abschnitt 5 fasst das Paper zusammen und weist auf weitere Schritte in der Entwicklung von SWAT hin.

2 SWAT Grundlagen und Funktionalität

SWAT berücksichtigt Prozesse in zwei Ausprägungen: entweder in Form von Petri-Netzen (Soll-Modell) oder in Form von Prozesslogs, welche ausgeführte Prozessinstanzen beinhalten.

2.1 Überblick und Bausteine von SWAT

Abbildung 1 gibt einen Überblick der einzelnen Bausteine von SWAT. SWAT verwendet *Petrinetze*, *Muster* oder *Logs* als Eingabe. Neben klassischen Petrinetzen können auch gefärbte Petrinetze, Workflownetze und Informationflusssnetze verarbeitet werden [SB13]. SWAT unterstützt den PNML-Standard als Eingabe- und Ausgabe-Format für Petrinetze. *Muster* repräsentieren in SWAT Sicherheitsanforderungen. Sie können in Linearer Temporaler Logik (LTL) oder als Petrinetz-

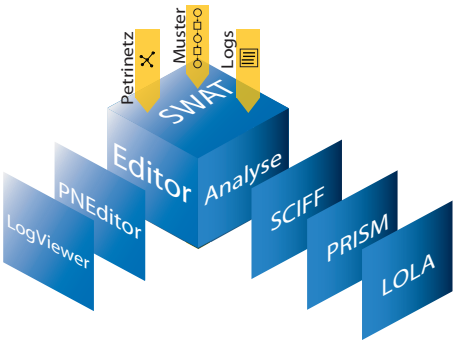


Abbildung 1: Bausteine von SWAT

[1] http://service-technology.org/lola/
[2] http://www.prismmodelchecker.org/
[3] http://lia.deis.unibo.it/research/sciff/

Fragmente definiert werden (siehe Abschnitt 2.3). *Logs* repräsentieren Protokolle von ausgeführten Workflows. SWAT unterstützt dabei gängige Log-Formate, wie MXML oder XES. SWAT erlaubt es zum einen durch *Editoren* die Petrinetze (*PNEditor*) zu bearbeiten und Logs zu visualisieren (*Logviewer*). Zum anderen bietet SWAT Verfahren zur *Analyse* dieser Eingaben an, wie *SCIFF*, *PRISM* oder *LoLA* (siehe Abschnitt 3).

Nachfolgenden werden die formalen Grundlagen von SWAT erläutert und auf die Modellierung von Geschäftsprozessen und Sicherheitsanforderungen eingegangen.

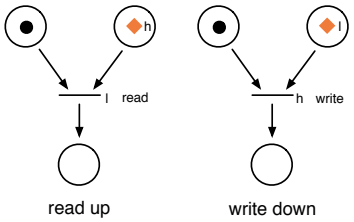
2.2 Formale Grundlagen und IFnet

Die zu Grunde liegende Methodik von SWAT basiert auf *Information Flow Nets* oder *Informationsflussnetzen* (IFnet). IFnet ist ein Petrinetz-Dialekt, der klassische Petrinetze um Konstrukte zur sicherheitsorientierten, formalen Notation von Workflows erweitert [SB13]. Petrinetze definieren die gültige Reihenfolge von Prozessaktivitäten mithilfe von Transitionen, die mit Plätzen verbunden sind. Vor- und Nachbedingungen für die Ausführung von Aktivitäten werden in Form von Regeln über die benötigte Anzahl von sog. Tokens angegeben. Diese Tokens befinden sich in den Plätzen des Petrinetzes und können durch das „feuern“ von Transitionen durch das Netz bewegt werden, indem sie aus Eingangs-Plätzen der betreffenden Transition „konsumiert“ und in Ausgangs-Plätzen derselben Transition „produziert“ werden. Die aktuelle Tokenkonfiguration eines Petrinetzes bestimmt dessen Zustand.

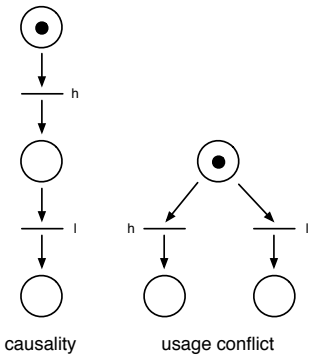
Klassische Petrinetze konzentrieren sich ausschließlich auf den Kontrollfluss eines Workflows, d.h. die Reihenfolge von Aktivitäten. Um gleichzeitig auch den Datenfluss zu berücksichtigen, der beschreibt, welche Aktivitäten welche Datenobjekte verwenden und mit welchem Modus zugegriffen wird [RW12, S.25], baut IFnet auf einer Definition für gefärbte Petrinetze auf, bei der unterschiedliche Tokentypen unterschieden werden können. Neben einem Typ für die Modellierung des Kontrollflusses, verwendet IFnet je einen Tokentyp für jede unterschiedliche Datenressource, die modelliert wird. Dabei kann es sich je nach Workflow um ein Dokument oder eine Variable handeln. Jede Transition wird gemäß ihrer Ressourcenverwendung entsprechend annotiert und die Methodik für das Bewegen von Tokens auf natürliche Art und Weise auf unterscheidbare Tokens erweitert. Transitionen können Information über das ausführende Subjekt (Person, Dienst, ...) enthalten. Im Hinblick auf sicherheitsorientierte Eigenschaften, erlaubt IFnet die Auszeichnung von Transitionen, Subjekten und Tokentypen, die für Datenressourcen stehen, mit einem Sicherheitslevel *High* und *Low*. Die mit *High* ausgezeichneten Elemente beschreiben den vertraulichen Bereich aus dem keine Information in den öffentlichen Bereich (alle mit *Low* ausgezeichneten Elemente) fließen darf. Diese informationsflussorientierte Sichtweise erlaubt eine überaus flexible und feingranulare Definition verschiedenster Sicherheitseigenschaften eines Workflows [Ros96]. Neben klassischen Eigenschaften der Zugriffskontrolle und Vertraulichkeit von Daten, lassen sich damit bspw. auch Anforderungen bzgl. der Vertraulichkeit von Prozessaktivitäten modellieren. Werden die vergebenen Level nicht als Sicherheits-, sondern als Integritätslevel aufgefasst, können durch geeignete Verfahren auch Integritätseigenschaften überprüft werden.

2.3 Muster in SWAT

Für die Identifikation von sicherheitskritischen (strukturellen) Eigenschaften eines Workflows verwendet SWAT Muster. Jedes Muster steht dabei für eine Schwachstelle, die es potentiellen Angreifern (externe Beobachter oder Prozessteilnehmer mit eingeschränkten Rechten) erlaubt, vertrauliche Informationen entweder direkt oder indirekt auf Basis von Rückschlüssen (Inferenzen) aus dem beobachtbaren Prozessverhalten zu erheben. Das kann z.B. eine fehlerhafte Konfiguration von Subjekt-Transition Zuordnungen sein, die es nicht autorisierten Prozessteilnehmern erlaubt auf direkten Weg an vertrauliche Information zu gelangen. Für solche Fälle hält SWAT Muster für das Bell-LaPadula [BL73] Sicherheitsmodell für Datenvertraulichkeit bereit (siehe Abbildung 2). Bzgl. Inferenzen verwendet SWAT Muster für verdeckte Kanäle, die sich in strukturellen Eigenschaften eines Workflows manifestieren und Informationsflüsse vom vertraulichen in den öffentlichen Bereich zulassen. Abbildung 2 zeigt beispielhaft zwei solcher Muster. Enthält ein Netz keines dieser beiden Muster, können bei dessen Ausführung keine Informationsflüsse auftreten, welche die Noninterference-Eigenschaft (S)BNDC verletzen.



(a) Bell-LaPadula Muster



(b) (S)BNDC Muster

Abbildung 2: Muster in SWAT

3 Analysemethoden von SWAT

Die zu Grunde liegende Idee der Muster-basierten Analyse in SWAT ist, dass der Nachweis für die Einhaltung spezifischer Sicherheitseigenschaften mittels Belegen für die Abwesenheit von entsprechenden Mustern bzw. deren Nicht-Ausführbarkeit innerhalb des Netzes erbracht wird. Aus diesem Grund werden die Muster auch als Anti-Patterns bezeichnet. Nachweise für die Abwesenheit von Mustern können auf unterschiedliche Art und Weise erbracht werden. Die anfängliche Version von SWAT nutzte alleinig den InDico Mechanismus [AW11]. Dabei wurde in einem statischen Schritt zunächst auf die Existenz eines Musters im zugrundeliegenden Petrinetz geprüft. Im positiven Fall erfolgte eine dynamische Analyse, die die Erreichbarkeit dieses Musters prüfte, d.h. die Möglichkeit während der Ausführung tatsächlich auf dieses Muster zu stoßen und es „auszuführen“. Dafür hat InDico den gesamten Erreichbarkeitsgraph (Zustandsraum des Netzes) generiert und diesen vollständig traversiert. Aufgrund der oft geringen strukturellen Komple-

xität von Workflows stellt dies in vielen Fällen zwar kein allzu großes Problem dar, kann die Effizienz und Abarbeitungsgeschwindigkeit jedoch so weit minimieren, dass bei on-the-fly Änderungen am IFnet die Analyse Ergebnisse erst nach gewisser Verzögerung zur Verfügung stehen. Um dennoch besonders komplexe Workflows zu bearbeiten wurde bei der aktuellen Version von SWAT Wert auf alternative, deutlich effizientere Verfahren gelegt. So ist es beispielsweise möglich, für große Workflows PRISM einzusetzen, einen hoch optimierten Model Checker, der in vielen Fällen den zu durchsuchenden Zustandsraum reduzieren kann. Speziell zur effizienten Analyse von Logdaten eines Workflows lässt sich SCIFF nutzen. SCIFF ist ein Verfahren, um aus einem gegebenen Verhalten zu prüfen, ob gegebene Regeln eingehalten wurden. Workflows, deren ursprüngliches Modell nicht vorhanden oder korrekt ist, lassen sich somit *a posteriori* auf Konformität oder Sicherheitsanforderungen hin prüfen. Ein auf Petri-Netze spezialisierter Model Checker ist LoLA ("Low Level Petri-Net Analyzer"). LoLA erlaubt die Eingabe eines Petri-Netzes und bietet Verfahren zum Prüfen petrinetzspezifischer Eigenschaften, wie zum Beispiel Beschränktheit (engl. boundedness). Es ist auch möglich, Eigenschaften als CTL-Formel (Computational Tree Logic) zu spezifizieren. Im Folgenden werden die einzelnen Analyse-Module von SWAT (PRISM, SCIFF, LoLA) beschrieben und deren Funktion jeweils an einem Beispiel gezeigt.

3.1 Analyse mit PRISM

PRISM als Tool zur Modellprüfung erlaubt weitgehende Möglichkeiten bei der Suche nach Sicherheitsschwachstellen. PRISM ermöglicht es, aussagenlogische Formeln für die Suche nach Sicherheitsschwachstellen zu benutzen. Die gegebenen Formeln werden, wie in InDico, für den gesamten Zustandsraum des Workflows geprüft. Findet PRISM ein aktives Muster, bricht es seine Analyse jedoch nicht ab, sondern durchsucht den verbleibenden Zustandsraum. Das Verfahren von PRISM berechnet so zusätzlich das Verhältnis von gefundenen, aktiven Verletzungen zu den Ausführungszuständen des Workflows, die keine Verletzung des Musters aufweisen. Somit kann nicht nur die Aussage getroffen werden, ob das gegebene Muster erfüllt oder nicht-erfüllt ist, sondern ausgegeben werden, für wie viele mögliche Zustände die Schwachstelle aktiv ist. Daraus lässt sich ableiten, wie wahrscheinlich ein Auftreten der gefundenen Schwachstellen bei der Ausführung des Workflows ist.

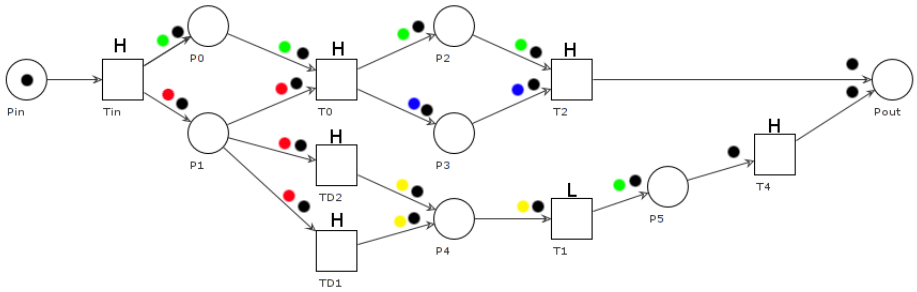


Abbildung 3: Beispielnetz für Analyse durch PRISM

Abbildung 3 enthält ein IFnet dass hier als Beispiel für die Analyse durch PRISM dient. Ein IFnet wird automatisiert in ein PRISM-Modell überführt, bei welchem es sich um eine mathematische Darstellung des IFnet als Formelmenge handelt. Die einzelnen Sicherheitsmuster müssen in linearer temporaler Logik (LTL) gestellt sein. Das Muster (b) aus Abbildung 2 lässt sich als LTL-Formel **High** $\diamond$ **Low** schreiben. Die Formel wird wahr und somit das Muster gefunden, wenn eine High-Aktivität existiert, auf die eine Low-Aktivität folgt. Die Formel wird für jeden möglichen Zustand des IFnet überprüft. Die Ausgabe von PRISM erfolgt in textueller Form (Abbildung 4):

```
P=?[F(P4_black>0 & P4_yellow>0 & TD2=0)]/P=?[F(P4_black>0 & P4_yellow>0)]:
  Result 0.5
```

Abbildung 4: Ausgabe der Analyse von PRISM

Dabei gibt die erste Zeile das zu suchende Muster aus: in diesem Fall das Verhältnis von Fällen, in denen der Platz P4 schwarze und gelbe Tokens beinhaltet und Transition TD2 inaktiv ist, zu den Fällen, in denen P4 Tokens enthält, aber der Zustand von TD2 irrelevant ist. Das Ergebnis der Analyse dieses Kausalitäts-Musters gibt die Häufigkeit dessen Verletzung an. In diesem Fall 0,5. Häufigkeitswerte sind hilfreich um die Stärke (und damit den Einfluss bestimmter Prozessaktivitäten auf potentiell unerwünschte Informationsflüsse) eines Kausalitätszusammenhangs zu bewerten.

3.2 Analyse mit SCIFF

SWAT kann zusätzlich SCIFF [AGL+05] und dessen Fähigkeit nutzen, Logdaten von Workflows auf gegebene Anforderungen zu überprüfen. SCIFF führt ein Verfahren, angelehnt an [FK97], aus und ermöglicht Analysen direkt auf Prozesslogs. Dieser Ansatz bietet den Vorteil, nicht auf die korrekte Umsetzung und Implementierung des Workflow-Modells für den produktiven Einsatz vertrauen zu müssen, sondern es kann der tatsächlich ”gelebte“ Prozess analysiert werden. Abbildung 5 zeigt beispielhaft einen Log-Eintrag eines Prozesslogs im MXML-Format. Hier sieht man, dass durch Pete die Aktivität „register request“ ausgeführt und dabei ein Datenelement „Costs“ verwendet wurde. Um Analysen durchzuführen, werden die Sicherheitsmuster, wie für die Analyse mit PRISM, in eine spezifische Sprache übersetzt. SCIFF nutzt eine eigene Syntax, die in [ACG+08] definiert ist. Mit dieser Syntax lassen sich analog zu einer LTL-Formel Regeln aufstellen, die ausdrücken, dass gewisse Aktionen in Zukunft abgearbeitet werden müssen oder nicht auftreten

```
<WorkflowLog>
 <Source program="Fluxicon_Nitro"/>
 <Process id="running-example-just-two-cases.mxml">
  <ProcessInstance id="1">
   <AuditTrailEntry>
    <Data>
     <Attribute name="Costs">50</Attribute>
    </Data>
    <WorkflowModelElement>register request</WorkflowModelElement>
    <EventType>complete</EventType>
    <Timestamp>2010-12-30T11:02:00.000+01:00</Timestamp>
    <Originator>Pete</Originator>
   </AuditTrailEntry>
   <AuditTrailEntry>
    <Data>
     <Attribute name="Activity">examine thoroughly</Attribute>
     ...
    </Data>
    ...
   </AuditTrailEntry>
   ...
  </ProcessInstance>
 </Process>
</WorkflowLog>
```

Abbildung 5: Beispiel Prozesslog für SCIFF (aus [VdAvdA11])

dürfen. Jede Instanz des Prozesslogs wird auf Einhaltung der gegebenen Ausdrücke überprüft. Abbildung 6 enthält ein Beispiel für eine in XML definierte Regel. Im Gegensatz zur *ex-ante* Analyse, welche Schwachstellen findet, die nicht zwingend bei der Ausführung des Workflows in Erscheinung treten müssen, sind die in einem Prozesslog gefundenen Sicherheitsverletzungen während der Ausführung des Workflows tatsächlich aufgetreten.

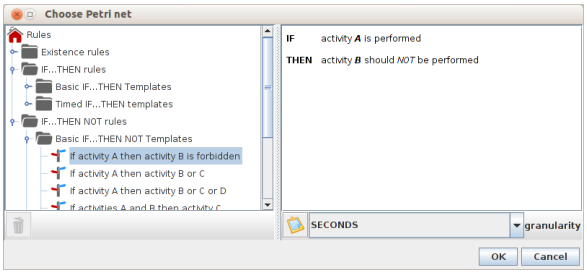


Abbildung 6: Beispiel einer Regel in SCIFF

3.3 Analyse mit LoLA

LoLA („Low Level Petri-Net Analyze“) ist ein speziell für Petri-Netze erstellter Model Checker. Dabei wird das Petrinetz in einer spezifischen Sprache modelliert (Abbildung 7). Die Umwandlung der Netze in dieses Format erfolgt durch SWAT automatisch. LoLA ermöglicht sowohl die Prüfung von allgemeinen Eigenschaften von Petrinetze (z.B. boundedness) als auch die Verwendung von CTL, um nach benutzerdefinierten Mustern zu suchen. Wie im obigen Absatz (3.1) erwähnt, müssen die Sicherheitsmuster aus Abbildung 2 zunächst transformiert werden. Das Ergebnis der Analyse trifft eine Aussage darüber, ob die gegebene Eigenschaft erfüllt ist oder nicht. LoLA gibt im Verletzungsfall Hinweise auf die Ursache in Form von kritischen Netzbereichen, die Verletzungen begünstigen.

```
PLACE p0;p1;p2;p3;...;

MARKING p7: 1 , p0: 1 , p14: 1 ;

TRANSITION ID_correct[H]
CONSUME p1 : 1 ;
PRODUCE p4 : 1 ;
...
TRANSITION Open[L]
CONSUME p14 : 1 , p11 : 1 ;
PRODUCE p12 : 1 ;
...
```

Abbildung 7: Das Beispiel aus Abbildung 9 in LoLA Syntax (gekürzt)

4 Realisierung als Standalone Applikation

Dieser Abschnitt stellt zunächst die Architektur von SWAT vor. Danach folgt eine Beschreibung der Anwendung und ihrer Benutzerschnittstelle. Schließlich wird das Arbeitsverzeichnis von SWAT erläutert.

4.1 Architektur

SWAT ist eine Java-basierte plattformübergreifende, modular aufgebaute Anwendung. Der Aufbau in Abbildung 1 stellt die Hauptanwendung (*SWAT*) und ihre Bestandteile (*Editor* und *Analyse*) vor.

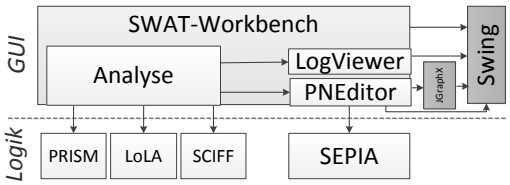


Abbildung 8: Architektur von SWAT

Abbildung 8 zeigt die Realisierung dieser Komponenten und ihre Unterteilung in Benutzeroberfläche (*GUI*) und *Logik*. Die vertikale Reihenfolge kennzeichnet Abhängigkeiten zwischen den beiden Schichten, wobei die Pfeile kennzeichnen, welche Komponente auf welche zugreift. Auf Grundlage von Java-Swing enthält die oberste Schicht (GUI) alle für die grafische Benutzeroberfläche zuständigen Komponenten. Die *SWAT-Workbench* ist für die Schaffung der Benutzerschnittstelle verantwortlich und bindet die nötigen Komponenten ein, um Dateien und Workflows bearbeiten und präsentieren zu können. Hier werden die konkreten Bausteine aus Abbildung 1 eingebettet:

(1) Der Petrinetz-Editor (PNEditor) basiert auf JGraphX, einem Swing-basierten Graph-Framework, erweitert dieses aber um Methoden zur Erstellung von Petrinetzen mithilfe des Security-oriented Petri Net Framework (*SEPIA*).

(2) Die Analyse-Komponente ist für die Einbindung und visuelle Repräsentation der angewendeten Analyseverfahren zuständig. Sie erlaubt es, sicherheitsorientierte Anforderungen und die zugehörige Netze so zu definieren, dass PRISM/ SCIFF dazu verwendet werden können, diese zu verifizieren.

4.2 Benutzerschnittstelle

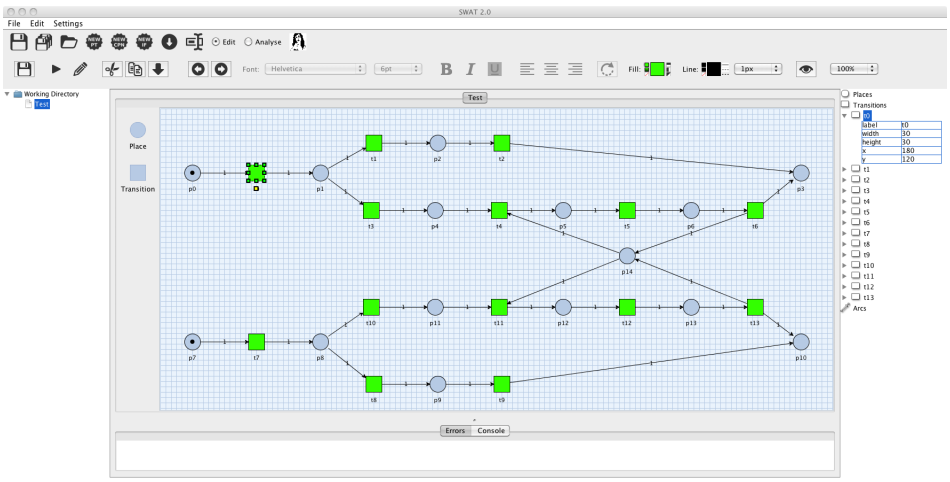


Abbildung 9: Aktueller Stand des SWAT Anwendungsfensters

Der Screenshot in Abbildung 9 zeigt die Anwendung nach dem Öffnen einer PNML-Datei. Der grafische Editor in der Mitte zeigt den geöffneten Workflow. Auf der linken Seite ist ein Baum mit Dateien zu sehen, die im aktuell gewählten Arbeitsverzeichnis zur

Verfügung stehen. Rechts neben dem grafischen Editor können die Eigenschaften der im Editor ausgewählten Elemente betrachtet und verändert werden.

SWAT unterscheidet zwischen dem Editieren und der Analyse von Workflows indem unterschiedliche Ansichten für diese Anwendungsfälle bereitgestellt werden. Wenn aus der Edit- zur Analyse-Ansicht gewechselt wird, wird die Eigenschafts-Leiste (rechter Rand) und der Datei-Explorer ausgeblendet und durch einen Analysebereich ersetzt, der die Auswahl und Anwendung von Analyseverfahren (PRISM, SCIFF, LoLA) ermöglicht.

4.3 Arbeitsverzeichnis

SWAT arbeitet auf einem gewählten Arbeitsverzeichnis im Dateisystem, welches alle Eingaben (Petrinetze, Muster, Logs) enthält. Dies erlaubt es, eine spezifische Eingabe-Konfiguration, welche sich aus mehreren Dateien zusammensetzt, zu speichern und wiederherzustellen. SWAT kann die Dateiformate PNML (Petrinetze und Muster), LTL (Muster), XML (High-Low-Labeling IFnet) und MXML/ XES (Logs) verarbeiten.

5 Zusammenfassung und Ausblick

Ziel dieses Beitrags ist es, SWAT vorzustellen. SWAT ist eine Plattform für die sicherheitsorientierte Analyse von Geschäftsprozessen. Sie soll Wissenschaftler und Prozessverantwortliche/-prüfer aus der Praxis dabei unterstützen, Geschäftsprozesse hinsichtlich der Einhaltung von (Sicherheits-) Eigenschaften zu überprüfen. Aktuelle Arbeiten an SWAT behandeln sowohl theoretische Aspekte als auch die Erweiterung des bisherigen Funktionsumfangs und Usability-Gesichtspunkte. Im Hinblick auf theoretische Fragestellungen, erforschen und erweitern wir den IFnetz-Ansatz, um es zu ermöglichen, weitere Sicherheitsfragen automatisiert zu analysieren, die bisher nicht in SWAT implementiert sind. Weitere Fragestellungen betreffen die Entwicklung effizienterer Analysealgorithmen für große Eingaben. Um SWAT flexibler einsetzbar zu machen wird daran gearbeitet, weitere Workflow-Formalismen zu unterstützten und als IFnet darstellbar zu machen.

Literatur

[Acc13] Rafael Accorsi. Sicherheit im Prozessmanagement. digma Zeitschrift für Datenrecht und Informationssicherheit, (2):72–76, 2013.

[ACG+08] Marco Alberti, Federico Chesani, Marco Gavanelli, Evelina Lamma, Paola Mello und Paolo Torroni. Verifiable agent interaction in abductive logic programming: the SCIFF framework. ACM Transactions on Computational Logic (TOCL), 9(4):29, 2008.

[AGL+05] Marco Alberti, Marco Gavanelli, Evelina Lamma, Paola Mello und Paolo Torroni. The SCIFF Abductive Proof-Procedure. In Stefania Bandini und Sara Manzoni,

Hrsg., AI*IA 2005: Advances in Artificial Intelligence, Jgg. 3673 of Lecture Notes in Computer Science, Seiten 135–147. Springer Berlin Heidelberg, 2005.

[ALS11] Rafael Accorsi, Lutz Lowis und Yoshinori Sato. Automated Certification for Compliant Cloud-based Business Processes. Business & Information Systems Engineering, 3(3):145–154, 2011.

[AW11] Rafael Accorsi und Claus Wonnemann. InDico: Information Flow Analysis of Business Processes for Confidentiality Requirements. In Jorge Cuellar et al., Hrsg., Security and Trust Management, Jgg. 6710 of Lecture Notes in Computer Science, Seiten 194–209. Springer, 2011.

[BL73] David Bell und Leonard LaPadula. Secure Computer Systems: Mathematical Foundations. MITRE Corporation, 1973.

[CGJ+09] Richard Chow, Philippe Golle, Markus Jakobsson, Elaine Shi, Jessica Staddon, Ryusuke Masuoka und Jesus Molina. Controlling data in the cloud: Outsourcing computation without outsourcing control. In Proceedings of the ACM Workshop on Cloud Computing Security, Seiten 85–90. ACM, 2009.

[DAC98] Matthew B. Dwyer, George S. Avrunin und James C. Corbett. Property Specification Patterns for Finite-State Verification. In Mark Ardis, Hrsg., Proceedings of the 2nd Workshop on Formal Methods in Software Practice: FMSP'98, Seiten 7–15, New York, 1998. ACM Press.

[FK97] Tze Ho Fung und Robert Kowalski. The IFF proof procedure for abductive logic programming. The Journal of logic programming, 33(2):151–165, 1997.

[LA11] Lutz Lowis und Rafael Accorsi. Finding vulnerabilities in SOA-based business processes. IEEE Transactions on Service Computing, 4(3):230–242, August 2011.

[MA13] Günter Müller und Rafael Accorsi. Why are business processes not secure? In Festschrift Buchmann, Jgg. 8260 of Lecture Notes in Computer Science, Seiten 240–254. Springer, 2013.

[MAHS10] Günter Müller, Rafael Accorsi, Sebastian Höhn und Stefan Sackmann. Sichere Nutzungskontrolle für mehr Transparenz in Finanzmärkten. Informatik Spektrum, 33(1):3–13, February 2010.

[Ros96] Bill Roscoe. Intensional Specifications of Security Protocols. In Proceedings of the 9th IEEE Computer Security Foundations Workshop, Seiten 28–38. IEEE Computer Society Press, 1996.

[RW12] Manfred Reichert und Barbara Weber. Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer, 2012.

[SB13] Thomas Stocker und Frank Böhr. IF-Net: A Meta-Model for Security-Oriented Process Specification. In Rafael Accorsi und Silvio Ranise, Hrsg., STM, Jgg. 8203 of Lecture Notes in Computer Science, Seiten 191–206. Springer, 2013.

[Sta11] Statistisches Bundesamt. Unternehmen und Arbeitstätten. Nutzung von Informations- und Kommunikationstechnologien in Unternehmen (in German). Statistisches Bundesamt, 2011.

[VdAvdA11] Wil MP Van der Aalst und Wil van der Aalst. Process mining: discovery, conformance and enhancement of business processes. Springer, 2011.

[WH10] Celia Wolf und Paul Harmon. The State of Business Process Management. BPTrends Report, 2010. Available at `http://www.bptrends.com/`.

Structural Composition Attacks on Anonymized Data

Tobias Nilges, Jörn Müller-Quade
Forschungsgruppe Kryptographie und Sicherheit
Institut für Theoretische Informatik
Karlsruher Institut für Technologie
Am Fasanengarten 5
76131 Karlsruhe
surname@kit.edu

Matthias Huber
Sichere Software und Kryptographie
FZI Forschungszentrum Informatik
Haid-und-Neu-Strae 10-14
76131 Karlsruhe
mhuber@fzi.de

Abstract: Anonymized releases of databases are increasingly available in public. The composition of two releases does not necessarily fulfil any anonymity notion, even if the individual releases do fulfil that anonymity notion on their own.

In this paper, we study composition scenarios and provide formalizations. We introduce a formal framework to study the composition of databases on a structural level and show the equivalence of the composition scenarios used in the literature. We show that known attacks on anonymity notions can be reduced to two simple properties and only need limited side information.
keywords: database anonymization, data composition, anonymity notions

1 Introduction

Anonymity has been in focus of research on privacy-preserving database disclosure [Swe02, Dwo06, FWCY10] over the past years. The goal is to publish an anonymized database in order to allow others to analyze it while preserving the privacy of the individuals represented in the database. For this scenario *anonymity notions* were introduced. Their purpose is to formally express the limitation of information disclosure from the database or guarantees provided by the anonymization. The property k-anonymity [SS98], for example, states that for any individual there have to be at least k (or zero) rows in the released database that can be associated with the individual.

A huge challenge is that the composition of two releases, i. e. the combination of two (independently) anonymized datasets, does not necessarily fulfil any anonymity notion [GKS08, Swe02]. Nevertheless, multiple releases from different sources as well as different releases from the same source are common in practice. This highlights the need to formally define

database composition in such a way that this problem can be considered in the design of anonymity notions. No formal investigation of database composition, however, has been carried out. Additionally, there are no results based on the structure instead of concrete attribute values.

Another challenge in privacy-preserving database disclosure is modeling background knowledge. Background knowledge (or side information) is knowledge an adversary obtains independently from other data sources. Background knowledge can be modeled as a database release. Therefore, many side-information attacks on the anonymity of database releases can be considered composition attacks. In our work, we show an interesting connection between modeling background knowledge and the composition of anonymized database releases. Because modeling background knowledge has proven to be difficult, it was only conducted for a small class of anonymization procedures [MKM$^+$07, CRL07, WFWP07]. Our framework gives way to new approaches for considering background knowledge.

Our Contribution. In this paper, we provide the first formal definition of database composition by extracting two scenarios from attacks presented in literature [Swe02, GKS08, NS08, MGKV06, LLV07]. We prove that these scenarios cover all possible forms of database composition and that they can be transformed into each other, thereby allowing for considering just one scenario during the design of anonymity notions.

We then move on to show that, to the best of our knowledge, most composition attacks (in fact all of the above mentioned attacks) can be reduced to two properties of anonymity procedures, namely *Locatability* and *Exact Sensitive Value Disclosure* (ESVD), first proposed by Ganta et al. [GKS08]. Locatability allows an attacker to determine a set of possible pre-images of an anonymized value including the correct value, while ESVD means that the anonymization procedure does not alter the sensitive values, i.e. keeps them as is in the anonymized database.

In order to control background knowledge in composition scenarios, we introduce the notion of symbolic databases. Although these databases are devoid of semantic information leaving merely the structure of the database, Locatability and ESVD still exist for symbolic databases. For some notions, no side information is necessary to carry out a composition attack. We show that ESVD suffices to break k-anonymity on a structural level. This provides evidence that notions implying Locatability and ESVD cannot provide an unconditional security guarantee if several anonymized databases are combined.

Related Work. In the context of database anonymization two lines of research have been established. On the one hand partition based anonymity notions such as k-anonymity [SS98], l-diversity [MGKV06] and t-closeness [LLV07], have been thoroughly analyzed, including analysis of the complexity of the methods [MW04, JA05, LDR05, ZYW05] and proposed attacks [XT06, LLV07, GKS08].

The other line of research examines adding noise to databases [PS86, Kim86, DP91, WdW96]. The main result is differential privacy [Dwo06]. There are several relaxations of this notion [MT07, DKM$^+$06, NRS07, MPRV09] as well as some variations [BBG$^+$11, MPRV09, KM12]. Recently, Gehrke et al. [GLP11] presented an attack against differential privacy.

Composition of anonymized databases has been investigated mainly in the context of differential privacy [DMNS06, DKM+06, GKS08, KM12], while partition based anonymity schemes are only explicitly considered by Ganta et al. [GKS08]. Nevertheless, attacks as presented by [MGKV06, LLV07, NS08] are also composition attacks.

Since differential privacy is secure under arbitrary background knowledge [Dwo06, GKS08], modeling background knowledge has been mainly examined in the area of partition-based anonymity schemes [MKM+07, CRL07, WFWP07].

Outline of the paper. This paper is structured as follows: In the next section, we define the basic concepts used throughout this paper. In Section 3, we introduce the concept of *symbolic databases* and present a framework around this notion. In Section 4, we present our main results: the equivalence of different composition scenarios and the interpretation of known attacks as specific exploits of two properties on a structural level. Section 5 concludes.

2 Preliminaries

In this section, we present definitions of the concepts used throughout this paper. After a formal definition of databases and quasi-identifiers we present three database composition scenarios. Then, we define generalizations of Locatability and Exact Sensitive Value Disclosure (cf. Ganta et al. [GKS08]).

2.1 Notation

Let a *database* $d = \{t_1, t_2, \ldots, t_n\}$ be a set of tuples with attributes $A = \{A_1, A_2, \ldots, A_m\}$. We denote the value of attribute A_i in tuple t_j by $t_j[A_i]$. Throughout this paper, we use the terms tuple and row interchangeably. We call the set of all i-th elements of each tuple in d the i-th column of d and $|d| = n * m$ the size of d.

Throughout this work, we use the operators projection π and selection σ as follows: Let $b = \{b_1, \ldots, b_l\} \subseteq A, l \leq m$, be a set of attribute values of d. Then, π projects a database d to a database d' that only contains a subset of the attributes of d:

$$\pi_b(d) = \cup_{j=1..n}(t_j[b_1], \ldots, t_j[b_l])$$

Likewise, σ selects a number of rows of d fulfilling some condition of the form:

$$\sigma_{a=v} = \{t \in d \mid t[a] = v\},$$

where $a \in A$ is an attribute and v is a value of the domain of A.

A set of non-sensitive attributes $Q = \{Q_1, \ldots, Q_w\} \subseteq A$ is called a *quasi-identifier* if these attributes serve to uniquely identify at least one individual in the database. Let $\mathsf{S} \subset A \setminus Q$ be the set of sensitive attributes, w.l.o.g. we only consider the case of $|\mathsf{S}| = 1$.

We abbreviate a probabilistic polynomial-time turing machine by PPT without explicitly describing the random input. In order to capture deterministic anonymization functions as well as probabilistic anonymization mechanisms, througout this paper, we talk about *anonymization mechanisms*.

2.2 Anonymity Notions

In order to describe the level of anonymity provided by an anonymization, anonymity notions have been defined. Many notions such as k-anonymity and notions based thereon describe the result of the anonymization mechanism. Notions like differential privacy describe the anonymization mechanism itself with respect to bounded [MPRV09] and unbounded [Dwo06] adversaries.

We stress that despite all anonymity notions seem to have a standard method in order to achieve them (e. g. adding noise for differential privacy, or to coarsen attribute values for k-anonymity), we have to distinguish between methods and guarantees. For example, another approach to achieve differential privacy is presented in [BBG$^+$11], without the addition of noise, but by assuming that the database already has some additional entropy for the adversary. On the other hand, the addition of Laplace noise does not yield differentially private methods in general, but only under certain assumptions about the distribution of attribute values.

In the following we assume an anonymity notion to state a form of guarantee $\mathcal{P}$, which can be achieved by a (probabilistic) anonymization mechanism f. We say f realizes $\mathcal{P}_k$, or $f \in \mathcal{P}_k$, where k is a privacy parameter or a set of privacy parameters.

2.3 Differential Privacy

We will briefly review the definition of differential privacy, since we will later use it to motivate our composition notions.

Definition 1 ([Dwo06]) *A randomized function f gives ε-differential privacy if for all data sets d_1 and d_2 differing on at most one element, and all $S \subseteq$ Range(f),*

$$\Pr[f(d_1) \in S] \leq \exp(\varepsilon) \cdot \Pr[f(d_2) \in S]$$

The intuition behind this is that an adversary cannot discern two databases (except for a small factor) that differs in a single entry, i.e. by the result of the mechanism it is nearly impossible to say if a certain entry is in the database or not. Differential privacy can be achieved by adding noise to the output, thereby hiding the true value. The noise distribution has to be Laplacian with variance $\frac{1}{\varepsilon}$.

2.4 Locatability and Exact Sensitive Value Disclosure

Locatability and Exact Sensitive Value Disclosure were first presented by Ganta et al. [GKS08]. While they use these notions only in the context of their intersection attack (cf. Section 2.5), we will present a generalized definition of their notions. Later, we will show how these properties translate to properties for symbolic databases and lead to an interesting type of attacks.

Definition 2 *Let f be an anonymization mechanism. Then we say f has* Locatability *iff there exists an adversary $\mathcal{A}$, a database d, a quasi-identifier Q and a tuple $t \in d$, such that $\mathcal{A}(\pi_Q(t))$ returns a strict subset p of $d' = f(d)$ with $f(t) \in p$.*

This definition of Locatability is weak in the sense that it can be very easy to achieve. Locatability only improves an attack if the distribution of sensitive values in the resulting partition differs from the distribution of sensitive values in the database. Consider for instance a database that has t-closeness [LLV07] with $t = 0$ for every sensitive attribute, i.e. the distribution of every partition is the same as in the complete database. Then Locatability is of no use to an adversary, however, achieving $t = 0$ in real databases is virtually impossible. In contrast, in the case of k-anonymity, an adversary can definitely identify the bucket of any individual [Swe02, GKS08].

The advantage of our generalization of Locatability from k-anonymity to general anonymity notions is that it enables us to highlight the similarity of many known composition attacks [GKS08, GLP11, NS08]. Additionally, we provide a precise formulation for identifying a set in the anonymized database that corresponds to an individual.

We now define Exact Sensitive Value Disclosure (ESVD).

Definition 3 *Let f be an anonymization mechanism. Then we say f has Exact Sensitive Value Disclosure iff at least one sensitive value of d is contained in $f(d)$. That is,*

$$\exists S \in \mathsf{S}, \exists v : v \in \pi_S(d) \wedge v \in \pi_S(f(d))$$

Recall that S describes the set of sensitive values. By removing the assumption that the anonymization mechanism is partition-based and possibly contains Locatability, we can prove that ESVD generally poses a threat during composition, even without background knowledge (cf. Section 4.4).

2.5 Intersection Attack

Algorithm 1 describes the intersection attack of Ganta et al. [GKS08]. As already mentioned, this attack heavily relies on ESVD and Locatability. First, the Locatability property is used to obtain a set of database entries associated with an individual of the overlapping population of the releases. Then, the sensitive values are extracted for each anonymized release. By intersecting these sets, one gets either the sensitive value associated with an

individual or a relatively small set of values. According to the measurements of Ganta et al., they can achieve a success rate of up to 60% based on real census data.

Algorithm 1 intersection attack (from [GKS08]).

1: $R_1, \ldots, R_n \leftarrow n$ independent anonymized releases
2: $P \leftarrow$ set of overlapping population
3: **for** each individual i in P **do**
4: **for** $j = 1$ to n **do**
5: $e_{ij} \leftarrow$ Get_equivalence_class(R_j, i)
6: $s_{ij} \leftarrow$ Sensitive_value_set(e_{ij})
7: **end for**
8: $S_i \leftarrow s_{i1} \cap s_{i2} \cap \ldots \cap s_{in}$
9: **end for**
10: **return** $S_1, \ldots, S_{|P|}$

In the algorithm, the function *Get_equivalence_class()* returns the partition of the database which contains the individual. By applying the function *Sensitive_value_set()* to the result of *Get_equivalence_class()* the sensitive values are extracted from the partitions.

In Section 4.3 we will show how this attack can be carried out with symbolic databases.

3 The Symbolic Framework

It has proven to be difficult to model or control arbitrary side information of an adversary in the context of database privacy [MKM+07, CRL07, WFWP07]. Our approach to this problem first eliminates all side information and later adds the information that cannot be hidden in the real world.

Our model is meaningful in the following sense: Successfully performing an attack in our model yields a successful attack in the real world. Thus, our ability to model ESVD and Locatability and to reduce attacks to these two properties in our framework also implies that the problems that arise from these properies are retained in the real world.

In this section, we first define the building blocks of our framework. To illustrate the relevance of this framework, in the following sections we show that (a) the composition notions presented in Section 4.1 can be proven as equivalent, (b) many composition attacks can be simulated in the symbolic framework (cf. [GKS08, GLP11, Swe02, NS08]) and, therefore, (c) do only need limited side information quantifiable in our model.

3.1 Representation of Databases

As a first step we define the class of structurally identical databases. In order to study database composition on a structural level, we replace all attribute values with meaning-

less symbols. This renders side information useless. Informally, structurally identical databases are a set of databases that can all be projected onto the same symbolic database.

Definition 4 $\mathfrak{S}(d) = \{d' | \exists w_r : w_r(d') = d\}$ *with* w_r *being a wrapper function that applies a bijective function* w_i *to each column* i *of* d*. We call* $\mathfrak{S}(d)$ *the databases* structurally identical *to* d*.*

$\mathfrak{S}$ implies an equivalence relation on databases. Instead of $d' \in \mathfrak{S}(d)$ we write $d \overset{\mathfrak{S}}{=} d'$ and call them structurally identical. Note that this notion also includes all permutations of a database, which is due to the definition of a database as a set.

Now, we can replace the values of any representative of $\mathfrak{S}(d)$ to create a symbolic database. The notion of symbolic databases is defined w. r. t. an adversary. In the following experiment, the adversary tries to relate a given symbolization of a database to one of two structurally identical databases she chose. The game is related to known security definitions for encryption schemes such as IND-CPA or IND-CCA2. In this experiment, we write $\leftarrow$ insted of $=$ in order to indicate that the allocation can involve a probabilistic mechanism:

Definition 5 *Let* $\mathcal{A}$ *be an adversary,* $d \in DB$*,* $i \in \{0, 1\}$*,* $\Sigma : DB \rightarrow DB$ *an injective function. The experiment* $Symb_{\mathcal{A}}^{i,\Sigma}(d)$ *is defined as follows:*

$d_0, d_1 \leftarrow \mathcal{A}(\mathfrak{S}(d))$
$d_0 \overset{\mathfrak{S}}{\neq} d_1$ *: return* $\perp$
$b \leftarrow \mathcal{A}(\Sigma(d_i))$
return $b \overset{?}{=} i$

Here Σ is a function that replaces the values of one of the two adversarially chosen databases with random symbols. This function is not known by the adversary. Then the adversary has to guess the pre-image of the symbolized database. With this experiment, we can define symbolic databases as follows:

Definition 6 *Let* Σ*:* $DB \rightarrow DB$ *be a bijective function. We call* Σ *a database symbolization function iff for all adversaries* $\mathcal{A}$ *and for all* $d \in DB$ *the following holds:*

$$\left| Pr[Symb_{\mathcal{A}}^{0,\Sigma}(d) = 1] - Pr[Symb_{\mathcal{A}}^{1,\Sigma}(d) = 1] \right| \leq \epsilon$$

for ϵ *negligible in* $|d|$*. We call* $\Sigma(d)$ *a* symbolic database *of* d*.*

Note that we use the definition above in order to define functions that change the attribute values of a given database rather than its structure.

In order to allow for composition scenarios, we use the same function that replaces the attribute values for corresponding columns in different databases. The definition of symbolic databases implies that attribute names get symbolized as well.

As an example for the symbolization, consider the databases depicted in Figure 1.

130**	< 30	*	AIDS
130**	< 30	*	Heart Disease
130**	< 30	*	Viral Infection
130**	< 30	*	Viral Infection
130**	≥ 30	*	Cancer
130**	≥ 30	*	Heart Disease
130**	≥ 30	*	Viral Infection
130**	≥ 30	*	Viral Infection

(a)

a	b	e	f
a	b	e	g
a	b	e	h
a	b	e	h
a	c	e	i
a	c	e	g
a	c	e	h
a	c	e	h

(b)

Figure 1: An example database (a) with attribute names omitted and a representative of its symbolic databases (b): attribute values are replaced with symbols. The two databases have identical structure, but the symbolic database is stripped of meaning. Therefore, no background knowledge can be used to extract information from it.

In this simplified example, values are mapped to letters of the alphabet. The replacement is done in no particular order because, according to our definition, the database in Figure 1(a) is actually a set. Another injective function that can serve as a symbolization function is a hash function.

Remark. It is still possible to code information into a symbolic database, e. g. how often a certain symbol occurs in the original database. We will use this property later in order to show the equivalence of the composition notions.

Remark. By requiring a polynomially bounded adversary in the above definitions, symbolization functions such as hash functions or encryption functions can be used.

3.2 Anonymization Scenarios of Symbolic Databases

There are different possibilities for applying symbolization during anonymization. Consider Figure 2, where a database d gets anonymized to d' with the anonymization mechanism f. The database symbolization Σ can be applied prior to the anonymization, after it, or both.

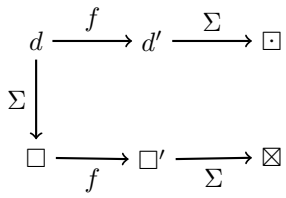


Figure 2: Anonymization scenarios for symbolic databases. d denotes the original database, f a database anonymization mechanism, Σ the database symbolization, and the different squares denote symbolic databases.

In general, even if the input of an anonymization mechanism is a symbolic database, this does not necessarily hold for the output. Consider for example an anonymization function that counts occurrences of attribute values. The output database then contains natural numbers that can be combined with background knowledge, e. g. that $3 > 2$. Thus we have to apply symbolization after anonymization.

We also want to apply the symbolization prior to anonymization for the following reasons: For one, this is the most general case for the examination of composition scenarios. If an anonymization does not compose in the most general case it also does not compose with additional knowledge. Another reason is that we do not want adversaries to exploit the Locatability oracle (i.e. use the oracle to win the game in Definition 5) which we define in the next section.

Therefore, we choose the following scenario: $d \xrightarrow{\Sigma} \square \xrightarrow{f} \square' \xrightarrow{\Sigma} \boxtimes$. We apply the symbolization Σ prior to anonymization and after it.

3.3 Locatability and ESVD for Symbolic Databases

We introduced Locatability and ESVD in Section 2.4. Since these properties can be exploited for composition attacks, in order to study composition attacks on a structural level, we need to be able to apply these properties to symbolic databases. In this section, we show how they translate into our framework.

Locatability. An adversary can always execute the anonymization mechanism f on a number of quasi-identifiers and compare the results to the anonymized database (although with new random coins, if the mechanism is probabilistic). This might enable an adversary to locate subsets in the anonymized database that relate to specific quasi-identifiers.

If we want an adversary to be able to do this with symbolic databases, we have to give her oracle access to $\Sigma \circ f$, where Σ is the symbolization and f the anonymization mechanism, since we do not want the adversary to know Σ. If f is probabilistic, the oracle has to use new random coins. Note that without the oracle, there is no Locatability for symbolic databases. Otherwise, the adversary would be able to invert Σ and thus win the game specified in Definition 5.

The notion of Locatability introduces a certain degree of side information into the symbolic framework, namely exactly the information that is necessary to obtain Locatability. However, the adversary still can not utilize her own background knowledge.

The construction of a Locatability oracle for an anonymity notion in the symbolic framework yields the background information necessary to perform Locatability for this notion in general, i. e. in the standard model. Thus the designer of an anonymity notion can check if the Locatability information is available in her setting.

ESVD. If an anonymization mechanism has the ESVD property, this property is kept in the symbolic framework, since we use one distinct symbolization for each column, and require that the same values in two corresponding columns are mapped to the same symbolic value.

With these definitions one can show that (a) the deanonymization of the Netflix dataset [NS08]

as well as (b) the attack of Gherke et al. [GLP11] is possible due to Locatability, (c) general attacks on k-anonymity presented in [Swe02] and in [GKS08] are possible due to ESVD and Locatability. These attacks can be simulated with symbolic databases where no other background knowledge than knowledge introduced by Locatability is available.

Locatability in the Netflix dataset stems from the sparsity of the columns representing movies, such that a set of movies identifies a set of individuals (cf. [NS08]). In the case of differentially private data [Dwo06], correlation between the entries can lead to Locatability. Gehrke et al. [GLP11] can locate an individual due to known associations with other individuals in the dataset and then derive the sensitive value. Although they do not use the term Locatability, their example is perfectly captured by our generalized definition.

In the next section, we will show a detailed example for an intersection attack with Locatability as well as one without Locatability.

4 Results in the Symbolic Framework

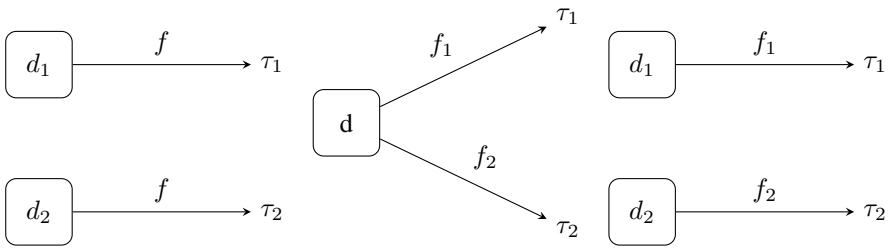
In this section we present our main results. We prove the equivalence of the composition scenarios (cf. Section 4.1) for symbolic databases, which translate straightforwardly to the plain model. Therefore, a proof of (non-)composability of an anonymity notion for a single composability scenario suffices for a general proof. Interestingly, due to the constructive nature of our proof, this also provides an adversary with a new attack. If her attack works out in one composition scenario, which was not considered for the anonymity notion, this attack can be converted by our technique to the other composition scenario.

We also show that known attacks such as the intersection attack of Ganta et al. (cf. Section 2.5) can be conducted with symbolic databases. We use a generalization of this attack and provide evidence that anonymization notions with ESVD cannot compose in general, even on a structural level with limited side information.

4.1 Composition Scenarios for Anonymized Databases

By analyzing the attacks presented in literature, e. g. [Swe02, GKS08, Dwo06, BBG$^+$11], we derive two composition schemes, both of which can be generalized to a third scheme. In the first composition scheme an adversary is given two excerpts of different databases, which are anonymized by the same anonymization mechanism (cf. Figure 3(a)). The second scheme represents the case where an adversary has access to two different anonymizations of the same database, which are anonymized with different anonymization mechanisms (cf. Figure 3(b)). The third approach is the direct generalization of both composition scenarios, where two (possibly) different tables are anonymized by two (possibly) different anonymization mechanisms. This is depicted in Figure 3(c). Obviously, this captures every possible form of composition.

We motivate our definitions for secure composition as follows. An adversary $\mathcal{A}$ tries to



(a) SC-ID (b) SC-IA (c) SC-IAD

Figure 3: The database composition notion *Secure Composition under Independent Anonymizations and Databases* (SC-IAD) and its two specializations *Secure Composition under Independent Databases* (SC-ID) and *Secure Composition under Independent Anonymizations* (SC-IA). In Section 4 we show that all three definitions are equivalent.

learn a predicate p of the database d. For two anonymizations $f_1(d_1)$ and $f_2(d_2)$ all the adversary can learn without combining both anonymizations is bounded by

$$\max_{i\in\{0,1\}} \{\Pr[\mathcal{A}(f_i(d_i)) = p(d_i)]\}$$

Thus we have to define the security of a composition of anonymized databases with respect to what an adversary can learn without composition. It is common knowledge that the release of anonymized data yields some information to an adversary, because perfect anonymization leads to no utility (cf. e. g. [Dwo06]). To capture this problem we introduce a *degradation function* κ which bounds the degradation of anonymity relative to the privacy parameter.

Example 1 *We illustrate our approach by applying it to differential privacy. The composition of differential privacy was shown among others in [DKM+06] and falls into the scenario SC-IAD. Here f_1 and f_2 represent anonymization mechanisms for $\mathcal{P}_{\varepsilon_1}$ and $\mathcal{P}_{\varepsilon_2}$, respectively. This example is highly simplified, but it illustrates the intuition behind our definitions. The following has to hold for secure composition (cf. Definition 9):*

$$\Pr[\mathcal{A}(f_1(d_1), f_2(d_2)) = p(d_1 \cup d_2)] \leq \kappa(\varepsilon_1, \varepsilon_2) \cdot \max_{i\in\{0,1\}} \{\Pr[\mathcal{A}(f_i(d_i)) = p(d_i)]\} \quad (1)$$

The composition of differential privacy yields the privacy parameter $\varepsilon_1 + \varepsilon_2$ for the left hand side of 1, while we can bound the other side by a privacy parameter $\max\{\varepsilon_1, \varepsilon_2\}$. The highest probability of an adversaries' guess can be substituted on both sides of the equation.

$$e^{\varepsilon_1+\varepsilon_2} \cdot 2 \max_{i\in\{0,1\}} \{\Pr[f_i(d_i) \in S]\} \leq \kappa(\varepsilon_1, \varepsilon_2) \cdot e^{\max\{\varepsilon_1,\varepsilon_2\}} \cdot \max_{i\in\{0,1\}} \{\Pr[f_i(d_i) \in S]\}$$

$$\Leftrightarrow \qquad 2e^{\varepsilon_1+\varepsilon_2} \leq \kappa(\varepsilon_1, \varepsilon_2) \cdot e^{\max\{\varepsilon_1,\varepsilon_2\}}$$

This leads to a degradation function $\kappa(\varepsilon_1, \varepsilon_2) = 2e^{\max\{\varepsilon_1,\varepsilon_2\}}$ and shows that the composition of differential privacy can be intuitively expressed by our definition.

We now state the definitions for the secure composition of the above mentioned scenarios.

Definition 7 *An anonymity notion $\mathcal{P}$ composes* securely under independent databases *(SC-ID) iff for all anonymization mechanisms $f \in \mathcal{P}_k$ there exists a degradation function κ such that for all adversaries $\mathcal{A}$, for all databases d_1 and d_2 and for all predicates $p : \{0,1\}^* \to \{0,1\}$ the following holds:*

$$\Pr[\mathcal{A}(f(d_1), f(d_2)) = p(d_1 \cup d_2)] \leq \kappa(k) \cdot \max_{i \in \{0,1\}} \{\Pr[\mathcal{A}(f(d_i)) = p(d_i)]\}$$

Of course, composition is only a privacy threat if the databases d_1 and d_2 have some sort of correlation, e. g. an overlap of individuals. By independent databases we mean that the two databases do not explicitly cover the same population. If a database anonymity notion $\mathcal{P}$ is a property of the anonymization mechanism (e. g. differential privacy), we have to extend the definition of $\mathcal{P}$ and quantify over pairs of anonymized databases instead of single databases.

Definition 8 *An anonymity notion $\mathcal{P}$ composes* securely under independent anonymizations *(SC-IA) iff for all anonymization mechanisms $f_1 \in \mathcal{P}_{k_1}$ and $f_2 \in \mathcal{P}_{k_2}$ there exists a degradation function κ such that for all adversaries $\mathcal{A}$, for all databases d and for all predicates $p : \{0,1\}^* \to \{0,1\}$ the following holds:*

$$\Pr[\mathcal{A}(f_1(d), f_2(d)) = p(d)] \leq \kappa(k_1, k_2) \cdot \max_{i \in \{0,1\}} \{\Pr[\mathcal{A}(f_i(d)) = p(d)]\}$$

Definition 9 *An anonymity notion $\mathcal{P}$ composes* securely under independent anonymizations and databases *(SC-IAD) iff for all anonymization mechanisms $f_1 \in \mathcal{P}_{k_1}$ and $f_2 \in \mathcal{P}_{k_2}$ there exists a degradation function κ such that for all adversaries $\mathcal{A}$, for all databases d_1 and d_2 and for all predicates $p : \{0,1\}^* \to \{0,1\}$ the following holds:*

$$\Pr[\mathcal{A}(f_1(d_1), f_2(d_2)) = p(d_1 \cup d_2)] \leq \kappa(k_1, k_2) \cdot \max_{i \in \{0,1\}} \{\Pr[\mathcal{A}(f_i(d_i)) = p(d_i)]\}$$

Obviously, if an anonymity notion composes securely under independent anonymizations and databases (SC-IAD), it also composes securely under independent databases (SC-ID) and under independent anonymizations (SC-IA). In Section 4.2, we will show that this also holds the other way round.

4.2 Equivalence of Composition Notions

We provide a formal proof that the composition scenarios defined in Section 4.1 can be transformed into each other. While this result seems to be rather intuitive if one considers normal databases, it appears to be difficult to prove the equivalence. Thus we show that these scenarios can be transformed if one considers symbolic databases, which implies a general transformation for standard databases.

Theorem 1 *For symbolic databases, the composition scenarios SC-IAD, SC-ID, and SC-IA are equivalent in terms of polynomial-time transformation, i. e. each instance of a composition scenario can efficiently be simulated with an instance of a composition scenario of another class.*

Proof 1 *In the following we use SC-ID, SC-IA, and SC-IAD as sets that contain all corresponding instances of database composition. Since the transformations from SC-ID and SC-IA to SC-IAD are trivial, we only need to show transformations from SC-IAD to SC-ID and SC-IA, respectively. Because a database can be extended, e. g. by adding blank columns and rows, in the following proof, w.l.o.g. we only consider databases of the same size.*

In our transformations, we encode additional information in the databases. For each bit of information we want to encode, we add a new column to the database. Since for symbolic databases, the actual attribute values have no meaning (otherwise an adversary would be able to invert the symbolization) and databases have no specific order (we defined them as sets), we encode information into the new columns by filling them with identical or different symbols.

SC-IAD $\subseteq$ ***SC-ID:*** *Let d_1, d_2 be databases and f_1, f_2 database anonymization mechanism of an SC-IAD instance. SC-ID instances require its two databases to be anonymized by the same anonymization mechanism f. We need to specify an algorithm $g : DB \times DB \to DB \times DB$ and an anonymization mechanism f with $f(g(d_1, d_2)[1]) = f_1(d_1)$ and $f(g(d_1, d_2)[2]) = f_2(d_2)$ where $[x]$ selects the x-th element of the preceding tuple.*

We design g such that it adds one additional column to both of its input databases. The column is filled according to the following scheme: For d_1, g fills the additional column of each tuple in d_1 with identical values. This encodes a 1. In the additional column of d_2, g fills all tuples except one with identical values. One row gets a different value, which encodes a 2. Consider for example the tables depicted in Figure 4. Here, g filled the

1	2
3	4

(a) d_1

5	6
7	8

(b) d_2

1	2	◯	5	6	□
3	4	◯	7	8	◯

(c) $g(d_1, d_2)$

Figure 4: An example for the two input databases d_1, d_2 of the algorithm g and its output $g(d_1, d_2)$. The Algorithm g adds an additional column to each database and encodes a 1 (equal symbols) into the additional column of first database and a 2 (one differing symbol) into the additional column of the second database.

additional column of d_1 with ◯, while the symbols used in the additional column of d_2 are □ and ◯.

With the information encoded in the additional columns, we can define a wrapper that applies the corresponding anonymization: Let $f(d'_1, d'_2)$ be a wrapper function for $(d'_1, d'_2) = g(d_1, d_2)$ that calls $f_1(d_i)$ for the database d'_i with a 1 encoded and $f_2(d_i)$ for the database d'_i with an encoded 2, respectively. We anonymize d_i instead of d'_i as this database does not contain the column which was added by g. The mechanisms f and g provide a transformation from SC-IAD to SC-ID.

Since f merely calls f_1 or f_2 and g adds an additional column to each of its input databases, the provided transformation from SC-IAD instances to SC-ID instances have an linear overhead in the size of the databases d_1 and d_2.

SC-IAD $\subseteq$ SC-IA: *Let d_1, d_2 be databases and f_1, f_2 database anonymization mechanisms of an SC-IAD instance. In order to transform a SC-IAD instance into an SC-IA instance, we need to specify an algorithm $h : DB \times DB \to DB$ and anonymization mechanisms f_1', f_2' with $f_1'(h(d_1, d_2)) = f_1(d_1)$ and $f_2'(h(d_1, d_2)) = f_2(d_2)$*

Consider an algorithm h that concatenates d_1 and d_2 tuple-wise and adds an additional column for every column in d_1 concatenated d_2. Into the first additional column, h encodes a 1 (see above) if the first column originates from d_1 and a 2 if the first column of the database is originally from d_2. In order to encode a 1, h chooses the same symbol for every entry in the additional column. In order to encode a 2, h chooses the same symbol for every entry in the additional column except for one entry where a different symbol is chosen. Consider for example Figure 5. Here, the first two tuples of the resulting database correspond to the first input database. Therefore, the first two additional columns only contain identical symbols, each.

1	2
3	4

(a) d_1

5	6
7	8

(b) d_2

1	2	5	6	◯	◯	□	□
3	4	7	8	◯	◯	◯	◯

(c) $h(d_1, d_2)$

Figure 5: An example for the two input databases d_1, d_2 of the algorithm h and its output $h(d_1, d_2)$. The algorithm h concatenates d_1 and d_2 tuple-wise and adds an additional column for each column. In each additional column, h encodes a 1 (identical symbols), if the corresponding column is from d_1 and a 2 (all but one symbols identical) if the corresponding column is from d_2.

With the information which column belongs to the first or the second database, we can define two algorithms, e. g. f_1' selects tuples belonging to the first database and anonymizes them with f_1. The mechanism f_i' discards each tuple in its input database, where the number in the corresponding column is different to i. Then it discards the extra columns and applies f_i to the result. Thus, f_1', f_2' and h allow a transformation from SC-IAD to SC-IA.

The algorithms f_i' execute a selection on the input database and then call f_1 or f_2. The resulting database of algorithm h contains all entries of the input databases and an additional column for each column of the input databases, such that the provided transformation from SC-IAD instances to SC-IA instances have an linear overhead in the size of the databases d_1 and d_2.

4.3 The Intersection Attack and Symbolic Databases

In Section 3.3, we have seen that ESVD and Locatability can be achieved in the symbolic framework. Therefore, the intersection attack (cf. Section 2.5, [GKS08]) can be performed. However, we stress that in general, Locatability is not even necessary for this attack, since it only decreases the size of the set of possible sensitive values.

In this section, we apply the intersection-algorithm to the example database we symbolized in Section 3.1. We use symbolic databases (cf. Figure 6) and we provide the adversary with some background knowledge in the form of a Locatability oracle. Instead of knowing the quasi identifiers of an individual (Alice) that occurs in both databases (as in a real scenario), we provide the adversary with the *symbolized* quasi identifiers of said individual.

a	b	e	f
a	b	e	g
a	b	e	h
a	b	e	h
a	c	e	i
a	c	e	g
a	c	e	h
a	c	e	h

(a) d_1

a	j	e	f
a	j	e	l
a	j	e	m
a	j	e	l
a	j	e	i
a	j	e	i
a	k	e	h
a	k	e	h

(b) d_2

Figure 6: Extended example of symbolic databases from Section 3.1. The first three columns denote the quasi identifiers while the last column denotes the sensitive value attribute. With these two databases, the Locatability oracle, and the symbolized quasi identifiers of an individual contained in both databases, an adversary is able to carry out the intersection attack.

The intersection attack can be carried out as follows: As a first step, the adversary uses the Locatability oracle in order to receive the buckets of the databases depicted in Figure 6 in which the sensitive value of Alice lies. The Locatability oracle for d_1 returns the set containing the sensitive values $\{f, g, h\}$, and the Locatability oracle for d_2 returns the set containing the sensitive values $\{f, l, m\}$, respectively. The adversary is able to determine that Alice has condition f by calculating the intersection of the two sets $\{f, g, h\} \cap \{f, l, m, \} = f$. This example shows that no background knowledge other than what is necessary to achieve Locatability is provided for a successful attack.

4.4 A General Attack on ESVD

In the previous section, we showed that the intersection attack can be carried out with symbolic databases. We now show that a generalization of the intersection attack with no side information at all can still lead to a disclosure of sensitive information.

Proposition 1 *Exploiting Exact Sensitive Value Disclosure is possible even without background knowledge about the attribute values.*

We now construct an example that serves as a witness for the symbolic non-composability of an anonymity notion that implements ESVD:

Proof 2 *Consider the database d depicted in Figure 7 and a mechanism f that partitions the database into buckets of size 2, calculates the product of each bucket with itself and*

a	1
b	2
c	3
d	4

Figure 7: An example database. It contains four tuples. $\{a, b, c, d\}$ are quasi-identifiers, $\{1, 2, 3, 4\}$ are sensitive attributes. Note that all entries are symbolic values. In particular, they cannot be added or multiplied. In our example, this database gets anonymized into the databases in Figures 8(a) and 8(b).

a	1
a	2
b	1
b	2
c	3
c	4
d	3
d	4

(a) Database d_1

a	1
a	3
c	1
c	3
b	2
b	4
d	3
d	4

(b) Database d_2

Figure 8: Two anonymized databases d_1 (a) and d_2 (b) that individually adhere 2-anonymity. The original database (cf. Figure 7) is anonymized by creating buckets of size 2 and joining each bucket with itself by its quasi-identifier (first attribute). The Database d_1 differs from d_2 because of different initial buckets in the original database. If d_1 and d_2 are intersected, the original database can be reconstructed.

joins the results. The results of f adhere 2-anonymity: Let the values in the first column be quasi-identifiers and the values in the second column be sensitive information. Then every quasi-identifier occurs at least two times in every database, making the databases 2-anonymous. Intersecting the databases in Figure 8, however, yields the original database of Figure 7, and intersecting symbolized versions of the anonymized databases yields a database structurally identical to the original database. This is possible due to the ESVD property of the anonymization mechanism used.

As shown by this example, ESVD can enable composition attacks on a structural level even if the adversary has no side information at all.

5 Summary and future work

In this paper, we presented a framework for studying the composition of anonymized database releases with no or very limited background knowledge. We provided a formal definition of composition notions and proved their equivalence. We provided strong evidence that Locatability end ESVD are the only necessary properties to enable composition

attacks by simulating them in our framework. We stress that all examples of composition attacks we gave are based solely on the structure of the databases and the Locatability Oracle we provided to the adversary, not on the meaning of its contents.

A possible extension of our framework is to model additional side information, e. g. relations of symbols. This is a natural extension, since values related in the real world are symbolized differently. We deem it significant to capture precisely the ratio of background information needed versus knowledge extracted from a release.

We also want to use the notion of symbolic databases in order to define anonymity notions more resilient to attacks involving background knowledge and composition. We propose to improve anonymity notions by including coarsening or aggregation of sensitive values [TZ11], similar to the idea of differential privacy. An important challenge for future work is to find tradeoffs between utility and privacy for methods that don't have undesirable properties like Locatability or Exact Sensitive Value Disclosure.

References

[BBG+11] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. Noiseless Database Privacy. In *ASIACRYPT*, pages 215–232, 2011.

[CRL07] Bee-Chung Chen, Raghu Ramakrishnan, and Kristen LeFevre. Privacy Skyline: Privacy with Multidimensional Adversarial Knowledge. In *VLDB*, pages 770–781, 2007.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *EUROCRYPT*, pages 486–503, 2006.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*, pages 265–284, 2006.

[DP91] G. Duncan and R. Pearson. Enhancing access to data while protecting confidentiality: prospects for the future. In *Statistical Science*, 1991.

[Dwo06] Cynthia Dwork. Differential Privacy. In *ICALP (2)*, pages 1–12, 2006.

[FWCY10] Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.*, 42(4), 2010.

[GKS08] Srivatsava Ranjit Ganta, Shiva Prasad Kasiviswanathan, and Adam Smith. Composition attacks and auxiliary information in data privacy. In *KDD*, pages 265–273, 2008.

[GLP11] Johannes Gehrke, Edward Lui, and Rafael Pass. Towards Privacy for Social Networks: A Zero-Knowledge Based Definition of Privacy. In *TCC*, pages 432–449, 2011.

[JA05] Roberto J. Bayardo Jr. and Rakesh Agrawal. Data Privacy through Optimal k-Anonymization. In *ICDE*, pages 217–228, 2005.

[Kim86] J. Kim. A method for limiting disclosure of microdata based on random noise and transformation. In *Proceedings of the Section on Survey Research Methods for the American Statistical Association*, pages 370–374, 1986.

[KM12] Daniel Kifer and Ashwin Machanavajjhala. A rigorous and customizable framework for privacy. In *PODS*, pages 77–88, 2012.

[LDR05] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Incognito: Efficient Full-Domain K-Anonymity. In *SIGMOD Conference*, pages 49–60, 2005.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *ICDE*, pages 106–115, 2007.

[MGKV06] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-Diversity: Privacy Beyond k-Anonymity. In *ICDE*, page 24, 2006.

[MKM+07] David J. Martin, Daniel Kifer, Ashwin Machanavajjhala, Johannes Gehrke, and Joseph Y. Halpern. Worst-Case Background Knowledge for Privacy-Preserving Data Publishing. In *ICDE*, pages 126–135, 2007.

[MPRV09] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil P. Vadhan. Computational Differential Privacy. In *CRYPTO*, pages 126–142, 2009.

[MT07] Frank McSherry and Kunal Talwar. Mechanism Design via Differential Privacy. In *FOCS*, pages 94–103, 2007.

[MW04] Adam Meyerson and Ryan Williams. On the Complexity of Optimal K-Anonymity. In *PODS*, pages 223–228, 2004.

[NRS07] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, pages 75–84, 2007.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust De-anonymization of Large Sparse Datasets. In *IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

[PS86] M. Palley and J. Siminoff. Regression methodology based disclosure of a statistical database. In *Proceedings of the Section on Survey Research Methods for the American Statistical Association*, pages 382–387, 1986.

[SS98] Pierangela Samarati and Latanya Sweeney. Protecting Privacy when Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression. Technical report, CMU SRI, 1998.

[Swe02] Latanya Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10:557–570, October 2002.

[TZ11] Hongwei Tian and Weining Zhang. Extending *l*-diversity to generalize sensitive data. *Data Knowl. Eng.*, 70(1):101–126, 2011.

[WdW96] L. Willenborg and T. de Waal. *Statistical Disclosure Control in Practice*. Springer Verlag, 1996.

[WFWP07] Raymond Chi-Wing Wong, Ada Wai-Chee Fu, Ke Wang, and Jian Pei. Minimality Attack in Privacy Preserving Data Publishing. In *VLDB*, pages 543–554, 2007.

[XT06] Xiaokui Xiao and Yufei Tao. Personalized privacy preservation. In *SIGMOD Conference*, pages 229–240, 2006.

[ZYW05] Sheng Zhong, Zhiqiang Yang, and Rebecca N. Wright. Privacy-enhancing - anonymization of customer data. In *PODS*, pages 139–147, 2005.

Multi-LHL protocol

Marika Mitrengová

Faculty of Mathematics, Physics and Informatics
Comenius University
Mlynska dolina
842 48 Bratislava, Slovakia
mitrengova@dcs.fmph.uniba.sk

Abstract: We present a password-authenticated group key exchange protocol where each user has his/her own password. Advantage of such protocol is in short passwords, which can be easily memorized. On the other hand these protocols face the low password entropy. In the first part we define security model based on models of Abdalla, Fouque and Pointcheval and Bellare, Pointcheval, Rogaway. We construct the MLHL (Multi-LHL) protocol, which is based on the LHL protocol proposed by Lee, Hwang and Lee. However, the LHL protocol is flawed as pointed by Abdalla, Bresson, Chevassut and Choo, Raymond. We prove that our protocol is secure authenticated key exchange protocol with forward secrecy property and that the protocol is resistant against attacks on the LHL protocol.

1 Introduction

With the explosion of its size, Internet became a major communication channel among people. However, in its basis, Internet is an inherently insecure channel. The essential part of securing such channel is an exchange of cryptographically strong keys. People are notoriously bad at remembering long (pseudo)random sequences and thus the classical solution is to store the key on some device ($e.g.$ hard disk, smart card) and protect it with a user password. This is inconvenient because the medium holding the original key needs to be carried everywhere by the user.

Password authenticated key exchange (PAKE) protocols were designed to alleviate this issue. They require a human user to remember only a short (easily-memorable) secret password. This is the major advantage for mobile users who need to authenticate at various places. PAKE protocols are therefore an interesting alternative of public key cryptography (PKI), especially in environments where the PKI is hard to deploy. Because of their ability to distill low-quality user passwords to strong keys, PAKE protocols have received a lot of attention [BMP00, Ja96, GL01, KOY01].

Although the original idea of PAKE protocol EKE [BM92] was designed only for two participants, PAKE protocols can be used to authenticate multiple parties as well. The most important requirement is to require only a single password for the user. Solutions, where user has to remember one password per group of participants obviously does not

scale with human memory. Moreover, in the case when one of the participants is compromised, the whole group needs to choose a new password. Instead, the schemes with single password per user offer much better user experience. However, this comes at the cost of incorporating one party which will be trusted by everyone – a trusted server.

Security issues with PAKE protocols: As opposed to other cryptographic schemes, PAKE protocols contain one weak link in their security and that is the user password. Therefore, they must be guarded against a dictionary attack against a known dictionary $\mathcal{DICT}$ of all possible passwords. The dictionary attack comes in two flavours – online and offline. The protocol can be easily protected against online dictionary attacks by blocking the user access after some unsuccessful tries. On the other hand the off-line dictionary attacks can (and should) be prevented by the PAKE protocol itself.

Related work. The research on PAKE protocols started with the EKE (Encrypted key exchange) protocol based on Diffie-Hellman key exchange. EKE was proposed by Bellowin and Merritt in [BM92]. However, the paper provides only very informal proof of security. This original work spawned a lot of new research ideas.

Observing recent work, Bellare, Pointcheval and Rogaway conclude that although many new PAKE protocols are proposed, the theory is lagging behind. Therefore, they define a security model for PAKE protocols and prove the correctness of EKE. Boyko, MacKenzie and Patel [BMP00] proposed 2PAKE protocols called PAK and PAK-X. They defined a new security model based on the model of Shoup [27]. Security of PAK is proved in the random oracle model under decisional Diffie-Hellman assumption. PAK is extended to a protocol PAK-X. It is built on the idea of a server which owns a user password verifier and the client stores a plaintext password. The authors formally proved the security of PAK-X, even when the server is compromised.

Kwon, Jeong, Sakurai and Lee [Kw06] deal with a multi-party scenario with a trusted server where each participant owns a different password. The goal of their protocols PAMKE_1 and PAMKE_2 is a group authentication and they note that designing PAKE protocols with trusted but curious server is quite involved task. Trusted server means that the server performs protocol steps and do not manipulate data in a different way. Curious means, that the server is honest, but we do not want it to know the computed session key. Another group authentication protocol was proposed by Lee, Hwang and Lee in [LHL04]. The LHL protocol is however not secure as showed by Abdalla, Bresson and Chevassut in [ABC06] where they propose a new protocol secure against this attack. Choo [Ch06] suggested another attack on the LHL protocol.

In [Ab11] suggested construction that is secure in a common reference string, therefore it does not rely on any idealized model. They prove the security of construction in the universally composable framework.

Hao and Ryan [HR11] suggested a protocol, where two participants send ephemeral public keys to each other. Then they encrypt the password by juggling the public keys in a verifiable way.

Our contribution. We were inspired by the LHL protocol [LHL04]. However in [ABC06, Ch06] it is shown that this protocol is not secure. We propose a new PGAKE protocol based on the LHL and prove that this protocol is secure in a random oracle model and ideal cipher model under decisional Diffie-Hellmann assumption. The security model is adopted from [BM92, AFP05, BPR00, Kw06]. Our construction is secure against the attacks from [ABC06, Ch06]. Secondly, every participant has his own secret password (compared to the protocol suggested in [ABC06]) and because of this, there are no problems with adding a new participant and with compromising some participant. On the other hand, this requires a help of a server, which knows the password of each participant. When the server knows the passwords, it could try to learn the session key (because it is curious). Therefore we want to have a protocol in which server could not learn established session key from knowledge of passwords and the communication it sees. Our main contribution is the proof of security (denoted as AKE-fs, see Definition 8) of our protocol.

2 Preliminaries

In this section, we establish the most important notation. If you are familiar with the standard notation in cryptography, it should be safe to skip this section.

2.1 Basic definitions

Random choice of an element R from a finite set T where the element R is chosen uniformly is denoted as $R \xleftarrow{\$} T$. By $M_1 \parallel M_2$ we denote *concatenation* of two strings M_1 and M_2. Random oracle is a function $f : Y_1 \to Y_2$ uniformly chosen from the set $Func(Y_1, Y_2)$ of all functions with domain Y_1 and range Y_2. We say that Turing machine A has oracle access to Turing machine B if machine A can use B as a function. We denote this fact as A^B. Symbol $\perp$ represents undefined value.
A symmetric encryption scheme is denoted as $E = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ and message authentication code scheme (MAC) is denoted as $M = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrf})$. A tag τ is computed as $\tau = \mathrm{Mac}_k(Msg)$ for message Msg with use of key k.

2.2 Protocols and adversaries

A single execution of a protocol is called a session. The set of protocol participants is $\mathcal{C} \cup \mathcal{S}$, where $\mathcal{C} = \{P_1, P_2, \ldots P_n\}$ is set of clients and $\mathcal{S}$ is set of servers. For simplicity, we assume that $|\mathcal{S}| = 1$. Each client $P_i \in \mathcal{C}$ has a password pw_i called long-lived key (LL-key) and server S has a vector of clients passwords $\langle pw_{S,P_i} \rangle_{P_i \in \mathcal{C}}$ ($pw_i = pw_{S,P_i}$ for all $P_i \in \mathcal{C}$ in symmetric case, otherwise they are different in asymmetric case). The j-th instance of participant P_i is denoted as Π_i^j and $ID(P_i)$ is a unique identifier of participant P_i (analogously j-th instance of server S is denoted as Ψ^j). A group of participants

$P_{i_1}, P_{i_2}, \ldots, P_{i_k}$ is denoted as $Grp_{i_1,i_2,\ldots,i_k}$.

Definition 1. [BR95] A protocol is a triple $\mathcal{P} = (\Pi, \Psi, LL)$, where Π specifies how each client behaves, Ψ specifies how server behaves and LL specifies the distribution of long-lived keys.

Definition 2. An adversary is a probabilistic polynomial-time Turing machine with oracle access to several other Turing machines. Running time of an adversary $\mathcal{A}$ is the length of description of $\mathcal{A}$ plus the worst case running time of $\mathcal{A}$.

Let CON be a cryptographic construction (algorithm), $\mathcal{A}$ be an adversary and xxx be any problem on CON (such as collision resistance of hash function, or discrete logarithm in a group G). $\mathbf{Adv}^{\text{xxx}}_{CON,\mathcal{A}}$ is a measure of adversary's advantage defined as a probability, that $\mathcal{A}$ succeeds to solve the problem xxx for CON. Sometimes, the advantage depends on some parameter, such as time of execution, length of the algorithm's input or the number of some queries. Let $\kappa_1, \kappa_2, \ldots \kappa_n$ be parameters needed for the security definition, then the adversary's advantage is denoted as $\mathbf{Adv}^{\text{xxx}}_{CONN,\mathcal{A}}(\kappa_1, \kappa_2, \cdots \kappa_n)$.

In this paper we adopt a Dolev-Yao model of an adversary, where the adversary intercepts whole communication during the execution of a protocol. The adversary can delay, change or deliver messages out of order, start a new execution of a protocol, acquire a LL-key of some participants and acquire a given session key. All abilities of the adversary are modelled through oracles defined in Section 3.

We use the notion of Parallel Decisional Diffie-Hellman assumption and a challenger $Chall^{\beta}(\cdot)$ defined by Abdalla et al. [ABC06] in our security proofs.

Definition 3 (Parallel Decisional Diffie-Hellmann assumption – PDDH$_n$)**.** Let G be a cyclic group of order q with generator g and $\mathcal{A}_D$ be an adversary (distinguisher). Two distributions are defined:
$PDH^*_n = \{(g^{x_1}, g^{x_2}, \ldots, g^{x_n}, g^{x_1x_2}, g^{x_2x_3}, \ldots, g^{x_nx_1}) | x_1, x_2, \ldots, x_n \stackrel{\$}{\leftarrow} Z^*_q\}$ and
$PDH^{\$}_n = \{(g^{x_1}, g^{x_2}, \ldots, g^{x_n}, g^{y_1}, g^{y_2}, \ldots, g^{y_n}) | x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n \stackrel{\$}{\leftarrow} Z^*_q\}$,
where $n > 2$. The PDDH$_n$ problem for input $(u_1, u_2, \ldots, u_n, w_1, w_2, \ldots, w_n)$ is to distinguish, from which distribution is it. The PDDH$_n$ assumption holds in a cyclic group G if and only if the advantage of every $\mathcal{A}_D$ on PDDH$_n$ problem in time t_{DDH} is negligible. This advantage is denoted as $\mathbf{Adv}^{\text{PDDH}_n}_{G,\mathcal{A}_D}(t_{DDH})$ and computed as:

$$\mathbf{Adv}^{\text{PDDH}_n}_{G,\mathcal{A}_D}(t_{DDH}) = |\Pr[\mathcal{A}_D(PDH^*_n) \to 1] - \Pr[\mathcal{A}_D(PDH^{\$}_n) \to 1]|.$$

In [ABC06], it was proved that for a group G, time t_{DDH}, an integer $n > 2$ and adversary $\mathcal{A}_D$ the PDDH$_n$ and DDH problems are equivalent in G:

$$\mathbf{Adv}^{\text{DDH}}_{G,\mathcal{A}_D}(t_{DDH}) \leq \mathbf{Adv}^{\text{PDDH}_n}_{G,\mathcal{A}_D}(t_{DDH}) \leq n \cdot \mathbf{Adv}^{\text{DDH}}_{G,\mathcal{A}_D}(t_{DDH})$$

$Chall^{\beta}(I)$ is an algorithm that on an input I outputs vectors from the distribution PDH^*_n, if the bit $\beta = 0$, otherwise it outputs vectors from the distribution $PDH^{\$}_n$. If the same I is given on the input again, then the same vectors are returned.

3 Security model

In this section we present a model based on [BM92], later extended in [BPR00] and adapted for group key exchange in [Kw06]. For identification of concrete session and instance of a partner in the session we defined notions *session identifier* and *partnering*.

Definition 4. A session identifier (sid) is a unique identifier of a session. It is the same for all participants in the session. The session identifier of the instance Π_i^j is denoted as sid_i^j. For the server instance Ψ^s is the session identifier denoted as sid^s.

If instances Π_i^j, Π_k^l and Ψ^s are in the same session, then $sid_i^j = sid_k^l = sid^s$.

Definition 5. A partner identifier pid_i^j for the instance Π_i^j is set of all identifiers of instances with whom Π_i^j wants to establish a session key. Instances Π_i^j and Π_k^l are partners, if

- $sid_i^j = sid_k^l \neq \perp$
- $\Pi_i^j \in pid_k^l$ and $\Pi_k^l \in pid_i^j$

The adversary controls whole communication. He can stop sent message, send message Msg, deliver messages out of order and intercept communication. His abilities are modelled using the following oracles:

- Send(Π_i^j, Msg) – sends the message Msg to the instance Π_i^j in the session sid_i^j and returns a reply of Π_i^j (according to the execution of the protocol). This oracle query simulates an active attack of the adversary.
- Send(Ψ^s, Msg) – similarly to the $Send(\Pi_i^j, Msg)$. This oracle query sends the message Msg to the instance of the server Ψ^s in the session sid^s and returns a reply of Ψ^s.
- Execute$(Grp_{i_1,i_2,\ldots,i_k}, S)$ – this oracle starts execution of a protocol between participants $P_{i_1}, P_{i_2}, \ldots, P_{i_k}$ and the server S. The result is a full copy of messages sent during execution of the protocol. This query models a passive attack, where adversary eavesdrops the execution of the protocol.
- Reveal(Π_i^j) – if the instance Π_i^j has established session key sk, then the oracle returns sk else return $\perp$. This oracle models scenario of session key leakage.
- Corrupt(P_i) – this query returns the LL-key pw_i of the participant P_i. This oracle models forward secrecy. (Such definition of Corrupt query is in a weak corruption model. In a strong corruption model Corrupt(P_i) returns an internal state of all instances of the participant P_i too.)
- Test(Π_i^j) – This query can be used only on a fresh/fs-fresh instance (see Def. 6). First a random bit $b \xleftarrow{\$} \{0,1\}$ is chosen. If instance Π_i^j has not established a session key sk, then $\perp$ is returned. If $b = 0$, then the real session key sk is returned else (if $b = 1$) random string $sk' \xleftarrow{\$} \{0,1\}^{|sk|}$ is returned.

Definition 6 (Fresh and fs-fresh instance)**.** The instance Π_i^j is *fresh*,

1. if oracle query Reveal was not made on the instance Π_i^j and its partners,
2. and if Corrupt query was not made on any protocol's participant in any session.

The instance Π_i^j is *fs-fresh*,

1. if oracle query Reveal was not made on the instance Π_i^j and its partners,
2. and if Corrupt query was not made on any protocol's participant in any session before Test query or Send query was not made on the instance Π_i^j.

Forward secrecy is security feature of a protocol and it is defined by Corrupt queries on the protocol. Informally, the protocol has forward secrecy property, if and only if revealing of LL-keys does not compromise previous established session keys.

Definition 7. Advantage $\mathbf{Adv}_{\mathcal{P},\mathcal{A}}^{\mathrm{AKE}(-\mathrm{fs})}(\kappa)$ of an adversary $\mathcal{A}$ attacking a protocol $\mathcal{P}$ in aforementioned model without (with) forward secrecy with security parameter κ is defined by a following game:
GameAKE(-fs)$_{\mathcal{P},\mathcal{A}}$:

- $\mathcal{A}$ can ask queries to Send, Reveal, Execute (and Corrupt in case of forward secrecy) oracles multiple times.
- Test query can be asked only once by $\mathcal{A}$ and only on a fresh (fs-fresh) instance.
- $\mathcal{A}$ returns a bit b'.

Let $Succ$ denote the event, that $b = b'$, where b is the bit randomly chosen during Test oracle. Then $\mathbf{Adv}_{\mathcal{P},\mathcal{A}}^{\mathrm{AKE}(-\mathrm{fs})}(\kappa) = |2 \cdot \Pr[Succ] - 1|$.

Definition 8. We say a protocol $\mathcal{P}$ is AKE (AKE-fs) secure multi-party PAKE protocol without (with) forward secrecy, if for all adversaries $\mathcal{A}$ running in polynomial time holds:

- all participant instances which are partners have the same session key,
- $\mathbf{Adv}_{\mathcal{P},\mathcal{A}}^{\mathrm{AKE}(-\mathrm{fs})}(\kappa) \leq \frac{Q(\kappa)}{|\mathcal{DICT}|} + \varepsilon(\kappa)$, where $\varepsilon(\kappa)$ is negligible and $Q(\kappa)$ denotes the number of on-line attacks (all Send queries to clients, server S and all Corrupt queries). $\mathcal{DICT}$ is a set of all possible passwords.

4 Our protocol

Our design goals for the new protocol are following:

- Enable group-based authentication with a *distinct password per user*. This however requires a presence of a *trusted* server.
- Protection against the previously mentioned attacks.

We meet both these design goals by replacing the first step of the LHL protocol with a secure communication through the trusted server. Because of this secure communication, the attacker can no longer exchange user identities by switching messages.

Similarly to LHL, our protocol works with a cyclic group G. We will use two pseudorandom hash functions $\mathcal{H}$ and $\mathcal{H}'$. New is the presence of a trusted server. Every participant P_i has password $pw_i \in \mathcal{DICT}$, which is shared with the server. To establish a secure connection to the server, we use arbitrary secure 2PAKE protocol denoted as 2P. We assume a symmetric encryption scheme modeled as an ideal cipher $E = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ and an existentially unforgeable under an adaptive chosen-message attack secure message authentication scheme $M = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrf})$.

Protocol MLHL (Multi-LHL):

1. Each participant P_i establishes a key sk_i with the server S using 2P protocol.
2. Establish a temporary key K_i between each pair of neighbours:
   (a) Each participant P_i chooses a random x_i, computes $z_i = g^{x_i}$ and sends message $P_i \to S : ID(P_i)||z_i^* = \mathcal{E}_{sk_i}(z_i)$. to the server
   (b) Server decrypts z_i^* and sends following messages to the participants P_{i-1} and P_{i+1}:
   $S \to P_{i-1} : ID(S)||ID(P_i)||\mathcal{E}_{sk_{i-1}}(z_i)$
   $S \to P_{i+1} : ID(S)||ID(P_i)||\mathcal{E}_{sk_{i+1}}(z_i)$
   (c) Each P_i decrypts received messages to obtain values z_{i-1} and z_{i+1} and computes $K_i = \mathcal{H}(z_{i+1}^{x_i})$, $K_{i-1} = \mathcal{H}(z_{i-1}^{x_i})$.
3. Each participant P_i computes $w_i = K_{i-1} \oplus K_i$, then he computes MAC $\tau_i = \mathrm{Mac}_{K_i}(ID(P_i)||w_i)$ and broadcasts message $(ID(P_i)||w_i||\tau_i)$.
4. When P_i receives messages $(ID(P_j)||w_j||\tau_j)$ from all other participants, he computes $K_j = \mathcal{H}(g^{x_{j-1}x_j})$ for all $j \in \{1, ..., n\}$ using the values w_j and K_{i-1}, in direction to the left (from $K_{i-1}, \ldots, K_n, \ldots, K_{i+1}, K_i$). During this computation, he verifies for received values $ID(P_j)$ and w_j their tags τ_j. For example, he starts with computing $K'_{i-2} = w_{i-1} \oplus K_{i-1}$, $\mathrm{Vrf}_{K_{i-1}}(ID(P_{i-1})||w_{i-1}, \tau_{i-1})$ and ends with $K'_i = w_{i+1} \oplus K_{i+1}$, $\mathrm{Vrf}_{K_{i+1}}(ID(P_{i+1})||w_{i+1}, \tau_{i+1})$. If all tag values are correct, then P_i continues with the next step, otherwise terminates.
5. P_i computes the session key $sk = \mathcal{H}'(K_1 \| K_2 \| \ldots \| K_n)$.

4.1 Security of MLHL protocol

Let G be a cyclic group with a generator g, for which the DDH assumption holds. Let $\mathcal{H}$ and $\mathcal{H}'$ be modeled as random oracles, where $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{l_\mathcal{H}}$ and $\mathcal{H}' : \{0,1\}^* \to$

$\{0,1\}^{l_{\mathcal{H}'}}$. Let 2P be an arbitrary secure 2PAKE protocol with length of the session key l_k, let $E = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ be symmetric encryption scheme defined as $\mathcal{E} : G \times \{0,1\}^{l_k} \to G$, $\mathcal{D} : G \times \{0,1\}^{l_k} \to G$ and modeled as an ideal cipher. Let $M = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrf})$ be an existentially unforgeable under adaptive chosen-message attack secure message authentication scheme. Symbol ε denotes a negligible function, $q_{\mathcal{E}}$ number of encryption queries, $q_{\mathcal{D}}$ number of decryption queries, $q_{send}, q_{execute}, q_{reveal}$ is number of Send, Execute, Reveal queries the attacker makes in underlying 2P protocol during the GameAKE-fs$_{\text{MLHL},\mathcal{A}_{\text{MLHL}}}$. Polynomial $p(\cdot)$ denotes the number of instances of the protocol MLHL executed through the Execute oracle or through the sequence of Send queries. Symbol $\mathcal{A}_X$ denotes adversary attacking construction X on its security property. Running times of adversaries $\mathcal{A}_{\mathrm{MLHL}}$, $\mathcal{A}_{\mathrm{2P}}$, $\mathcal{A}_M$ and $\mathcal{A}_{\mathrm{DDH}}$ are denoted $t_{\mathrm{MLHL}}, t_{\mathrm{2P}}, t_M, t_{DDH}$ and κ is security parameter.

Theorem 1. *Assume that every participant P_i has a secret key $pw_i \in \mathcal{DICT}$, which is shared with the server S. We suppose, that the adversary $\mathcal{A}_{\mathrm{MLHL}}$ establishes $p(\kappa)$ sessions during the GameAke$_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}}}$ between n participants for some polynomial $p(\cdot)$. Then the advantage of the adversary $\mathcal{A}_{\mathrm{MLHL}}$ in attacking the protocol* MLHL *is*

$$\begin{aligned}\mathbf{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}}}(\kappa, t_{\mathrm{MLHL}}) \leq 2\Bigg(&\frac{3(q_{\mathcal{E}}+q_{\mathcal{D}})^2}{2|G|} + \frac{3p(\kappa)\cdot n\cdot q_{\mathcal{D}}}{2^{l_k}} \\ &+ p(\kappa)\cdot n\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{2P},\mathcal{A}_{\mathrm{2P}}}(t_{\mathrm{2P}}, q_{execute}, q_{send}, q_{reveal}) + 2\varepsilon \\ &+ \frac{np(\kappa)^2}{2^{l_k+1}} + 5p(\kappa)\cdot n\cdot \mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}) + 8q_{\mathcal{E}}/2^{l_k} \\ &+ 4\mathbf{Adv}^{\mathrm{MAC-forge}}_{M,\mathcal{A}_M}(t_M)\Bigg).\end{aligned}$$

Looking at the definition of fs-fresh instance on which an adversary makes a Test query we have following cases of Corrupt query usage (on instance in Test query) during the game GameAKE-fs$_{\text{MLHL},\mathcal{A}_{\text{MLHL}}}$:

- $Case_1$: No Corrupt query was made during the execution of the game GameAKE-fs$_{\text{MLHL},\mathcal{A}_{\text{MLHL}}}$. In this case the adversary can ask Send, Execute and Reveal queries.
- $Case_2$: In this case, there must be at least one Corrupt query and all Corrupt queries were made after a Test query in the game GameAKE-fs$_{\text{MLHL},\mathcal{A}_{\text{MLHL}}}$ (note that in this case the session key was established for instance on which Test query was made). Here are allowed Send, Execute and Reveal queries.
- $Case_3$: In this case, there must be at least one Corrupt query and some Corrupt query was made before a Test query in the game GameAKE-fs$_{\text{MLHL},\mathcal{A}_{\text{MLHL}}}$. In this case, only Execute and Reveal queries are allowed, due to preservation of the fs-fresh property (adversary can not ask Send query on instances of other participants in the same session, because if he starts to ask Send queries in the session, he must ask Send queries on instance on which he will ask a Test query to finish the protocol execution correctly).

Therefore, we can divide advantage of the adversary attacking on AKE-fs security into advantage of the adversary in every of these cases:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}}}(t_{\mathrm{MLHL}}) &= \mathbf{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}},Case_1}(t_{\mathrm{MLHL}}) \\ &+ \mathbf{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}},Case_2}(t_{\mathrm{MLHL}}) \\ &+ \mathbf{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}},Case_3}(t_{\mathrm{MLHL}}).\end{aligned}$$

We prove the theorem for every case in three lemmas by sequence of games, starting with the game G_0 simulating the real protocol. In these games we simulate participants of the protocol and their behavior. By $Succ_i$ we denote that $b = b'$ in the game G_i, where b was randomly chosen bit in Test query and b' is the output of the adversary.
For simplicity we suppose, that the adversary asks Execute queries on group with the number of users n. Similarly when the protocol is simulated through $Send$ queries, we assume that the number of users is n too.

Proof. Due to space limitations, the full proof of theorem is in full version [Mi13]. The proof of the AKE security of the MLHL protocol is in Appendix A. □

4.1.1 Acknowledgement.

This paper was supported by VEGA grant number 1/0259/13 and by Comenius University grant number UK/407/2013.

References

[BM92] Steven M. Bellovin and Michael Merritt,Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks, In IEEE Computer Society Symposium on Research in Security and Privacy, pp. 72–84, IEEE Computer Society Press, 1992.

[LHL04] Lee, Su-Mi and Hwang, Jung Yeon and Lee, Dong Hoon, Efficient Password-Based Group Key Exchange, Trust and Privacy in Digital Business, First International Conference, TrustBus'04, pp. 191-199, LNCS 3184, Springer, 2004.

[ABC06] Michel Abdalla and Emmanuel Bresson and Olivier Chevassut, Password-based Group Key Exchange in a Constant Number of Rounds, Public Key Cryptography - PKC'06 - 9th International Conference on Practice and Theory in Public Key Cryptography, pp. 427–442, LNCS 3958, Springer, 2006.

[Ch06] Choo, Kim-Kwang Raymond, On the Security Analysis of Lee, Hwang & Lee (2004) and Song & Kim (2000) Key Exchange / Agreement Protocols, Informatica, 17, pp. 467-480, IOS Press, 2006.

[AFP05] Michel Abdalla and Pierre-Alain Fouque and David Pointcheval, Password-based authenticated key exchange in the three-party setting, PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography, pp. 65–84, LNCS 3386, Springer, 2005.

[BPR00] Mihir Bellare and David Pointcheval and Phillip Rogaway, Authenticated Key Exchange Secure Against Dictionary Attacks, Advances in Cryptology – EUROCRYPT'00, International Conference on the Theory and Application of Cryptographic Techniques, pp 139–155, LNCS 1807, Springer, 2000.

[Kw06] Jeong Ok Kwon and Ik Rae Jeong and Kouichi Sakurai and Dong Hoon Lee, Password-authenticated multiparty key exchange with different passwords, IACR Cryptology ePrint Archive, 2006.

[BR95] Mihir Bellare and Phillip Rogaway, Provably secure session key distribution: The Three Party Case, Proceedings of the twenty-seventh annual ACM symposium on Theory of computing, STOC '95, pp. 57–66, ACM, 1995.

[CPS08] Jean-Sébastien Coron and Jacques Patarin and Yannick Seurin, The Random Oracle Model and the Ideal Cipher Model Are Equivalent, Advances in Cryptology - CRYPTO'08, 28th Annual International, pp. 1–20, LNCS 5157, Springer, 2008.

[KL07] Katz, Jonathan and Lindell, Yehuda, Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series),Chapman & Hall/CRC, 2007.

[KOY03] Jonathan Katz and Rafail Ostrovsky and Moti Yung, Forward Secrecy in Password-Only Key Exchange Protocols, Security in Communication Networks, Third International Conference, pp. 29–44, LNCS 2576, Springer, 2003.

[BMP00] Victor Boyko and Philip Mackenzie and Sarvar Patel, Provably secure password-authenticated key exchange using Diffie-Hellman, Advances in Cryptology - EUROCRYPT'00, International Conference, pp. 156–171, LNCS, Springer, 2000.

[Ja96] David P. Jablon, Strong Password-Only Authenticated Key Exchange, SIGCOMM Computer Communication Review 26, pp. 5–26, ACM, 1996.

[Wu98] Thomas Wu, The secure remote password protocol, Proceedings of the Network and Distributed System Security Symposium, NDSS'98, The Internet Society, pp. 97–111, 1998.

[GL01] Oded Goldreich and Yehuda Lindell, Session-Key Generation using Human Passwords Only, Advances in Cryptology - CRYPTO'01, 21st Annual International Cryptology Conference, pp. 408–432, LNCS 2139, Springer, 2001.

[KOY01] Jonathan Katz and Rafail Ostrovsky and Moti Yung, Efficient Password-Authenticated Key Exchange using Human-Memorable Passwords, Advances in Cryptology - EUROCRYPT'01, International Conference on the Theory and Application of Cryptographic Techniques, pp. 475–494, LNCS, Springer, 2001.

[Mi13] Marika Mitrengová, Multi-LHL protocol, Cryptology ePrint Archive, Report 2013/621, `http://eprint.iacr.org/`, 2013.

[HR11] Feng Hao and Peter Ryan, Password Authenticated Key Exchange by Juggling,Security'08 Proceedings of the 16th International conference on Security protocols, pp. 159–171, Springer, 2011.

[Ab11] Michel Abdalla1 and Celine Chevalier and Louis Granboulan and David Pointcheval, Contributory Password-Authenticated Group Key Exchange with Join Capability,CT-RSA,pp. 142–160, LNCS, Springer,2011.

A Advantage of adversary in $Case_1$

In this section we prove the AKE-fs security of the MLHL protocol in $Case_1$, where the adversary does not make any Corrupt queries. If no Corrupt queries are made, it is sufficient to prove the AKE security instead of AKE-fs.

Lemma 1. *The advantage of the adversary from $Case_1$ is:*

$$\begin{aligned}\boldsymbol{Adv}^{\mathrm{AKE-fs}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}},Case_1}(t_{\mathrm{MLHL}}) \leq 2\Bigg(& \frac{(q_{\mathcal{E}}+q_{\mathcal{D}})^2}{2|G|} + \frac{p(\kappa)\cdot n\cdot q_{\mathcal{D}}}{2^{l_k}} \\ & + p(\kappa)\cdot n\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{2P}}(t_{\mathrm{2P}}, q_{execute}, q_{send}, q_{reveal}) \\ & + \frac{np(\kappa)^2}{2^{l_k+1}} + 2p(\kappa)\cdot n\cdot \mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}) \\ & + 2\mathbf{Adv}^{\mathrm{MAC-forge}}_{M}(t_M) + 4q_{\mathcal{E}}/2^{l-k}\Bigg).\end{aligned}$$

Proof. We start with the simulation of the real protocol.
Game G_0:
This is a game simulating the real protocol. From the definition 7 we have:

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}}}(t_{\mathrm{MLHL}}) = 2\Pr[Succ_0] - 1.$$

Because 2P could represent arbitrary secure 2PAKE protocol, without the loss of generality we assume that the protocol has l flows of messages. By sk_i =2P(P_i, S) we denote, that the key sk_i was computed with simulation of 2P between P_i and S. When a participant awaits more than one message, we denote it as a concatenation (see definitions of Send$_3$ and Send$_4$ oracles). In this game we simulate Send oracles as described bellow (we skip the description of Execute, Test and Reveal queries, because they are straightforward from their definition). The simulation of Send queries is divided into $l+4$ types of queries (l is number of messages sent during the 2P protocol). Such Send query represents concrete type of message, which was sent.

Send$^1_1(\Pi^j_i, Msg)$
simulate first step of the 2P protocol, message Msg of the form $(ID(P_{i_1})||ID(P_{i_2})||\ldots||ID(P_{i_{n-1}})||ID(S))$ is sent to the instance Π^j_i informing that the instance Π^j_i is going to establish a session key with participants $P_{i_1}, P_{i_2}, \ldots, P_{i_{n-1}}$,
return the message, which is the result of simulation of the first step of the 2P protocol.

$\vdots$

Send$_1^l(\psi^s, Msg)$
simulate the last step of the 2P protocol,
return the last message of the 2P protocol computed according to the rules of 2P.

Send$_2^1(\Pi_i^j, Msg)$
Msg is the last message sent by the server ψ to Π_i^j in 2P,
$sk_i =$2P(P_i, S),
$x_i \xleftarrow{\$} G$,
$z_i = g^{x_i}$, $z_i^* = \mathcal{E}_{sk_i}(z_i)$
return $(ID(P_i)||z_i^*)$

Send$_2^2(\psi^s, Msg)$
Msg has the form $(ID(P_i)||Msg')$
$z_i = \mathcal{D}_{sk_i}(Msg)$,
$z_{i-1}^{**} = \mathcal{E}_{sk_{i-1}}(z_i)$,
$z_{i+1}^{**} = \mathcal{E}_{sk_{i+1}}(z_i)$
return $(ID(S)||ID(P_i)||z_{i-1}^{**})$, $(ID(S)||ID(P_i)||z_{i+1}^{**})$

Send$_3(\Pi_i^j, Msg_{i-1}||Msg_{i+1})$
Msg_{i-1} and Msg_{i+1} have the form $(ID(S)||ID(P_{i-1})||Msg'_{i-1})$ and $(ID(S)||ID(P_{i+1})||Msg'_{i+1})$
$z_{i-1} = \mathcal{D}_{sk_i}(Msg_{i-1})$, $z_{i+1} = \mathcal{D}_{sk_i}(Msg_{i+1})$,
$K_{i-1} = \mathcal{H}(z_{i-1}^{x_i})$, $K_i = \mathcal{H}(z_{i+1}^{x_i})$,
$w_i = K_{i-1} \oplus K_i$, $\tau_i = \mathrm{Mac}_{K_i}(ID(P_i)||w_i)$
return $(ID(P_i)||w_i||\tau_i)$

Send$_4(\Pi_i^j, Msg_0||\ldots||Msg_{i-1}||Msg_{i+1}||\ldots||Msg_n)$
Msg_j has the form $(ID(P_j)||w_j||\tau_j)$,
$j \in \{0, \ldots, i-1, i+1, \ldots, n\}$,
if Vrf$_{K_{i-1}}(ID(P_{i-1})||w_{i-1}, \tau_{i-1}) = 1$
then $K_{i-2} = w_{i-1} \oplus K_{i-1}, \ldots$
if Vrf$_{K_{i+1}}(ID(P_{i+1})||w_{i+1}\tau_{i+1}) = 1$
then $K_i = w_{i+1} \oplus K_{i+1}$,
$sk = \mathcal{H}'(\mathcal{H}(K_1)\|...\|\mathcal{H}(K_n))$,
return "accept"
else if any of MAC verifications fails, **return** "terminated"

Game G'_0:
In this game we simulate encryption and decryption oracles. We work with a list $\Lambda_{\mathcal{E}}$ of tuples $(type, sid_i^j, i, \alpha, sk, z, z^*)$, where we store previous answers of encryption/decryption queries. $Type$ takes values enc/dec, sid_i^j is a session ID of the instance Π_i^j, α is value used in other games, sk is encryption/decryption key and $z^* = \mathcal{E}_{sk}(z)$. Moreover, we use a list $\Lambda_{2\mathrm{P}}$ of tuples (sid, i, sk) where we store previously established session keys sk_i in the 2P protocol in session sid for the participant P_i. We simulate encryption and decryption as follows:

- $\mathcal{E}_{sk}(z)$ – if $(\cdot,\cdot,\cdot,\cdot,sk,z,z^*) \in \Lambda_{\mathcal{E}}$, we return z^* otherwise we choose $z^* \xleftarrow{\$} G$, if $(\cdot,\cdot,\cdot,\cdot,sk,\cdot,z^*) \in \Lambda_{\mathcal{E}}$, we stop the simulation and the adversary wins (because such situation represents a collision). Otherwise we add a record $(enc,\perp,\perp,\perp,sk,z,z^*)$ to $\Lambda_{\mathcal{E}}$ and return z^*.
- $\mathcal{D}_{sk}(z^*)$ – if $(\cdot,\cdot,\cdot,\cdot,sk,z,z^*) \in \Lambda_{\mathcal{E}}$, we return z otherwise
  - if $(sid_i^j,i,sk) \in \Lambda_{2\mathrm{P}}$, we choose $z \xleftarrow{\$} G^*$, if $(\cdot,\cdot,\cdot,\cdot,sk,z,\cdot) \in \Lambda_{\mathcal{E}}$, we stop the simulation and the adversary wins. Otherwise we return z and add a record $(dec,sid_i^j,i,\perp,sk,z,z^*)$ to $\Lambda_{\mathcal{E}}$.
  - if $(sid_i^j,i,sk) \notin \Lambda_{2\mathrm{P}}$, we choose $z \xleftarrow{\$} G^*$, if $(\cdot,\cdot,\cdot,\cdot,sk,z,\cdot) \in \Lambda_{\mathcal{E}}$, we stop the simulation and the adversary wins. Otherwise we return z and add a record $(dec,\perp,\perp,\perp,sk,z,z^*)$ to $\Lambda_{\mathcal{E}}$

This game is the same as the previous unless:

- Collision occurs in the simulation of encryption/decryption. This event happens with probability $\approx \frac{(q_{\mathcal{E}}+q_{\mathcal{D}})^2}{2|G|}$, where $q_{\mathcal{E}}$ is a number of encryptions and $q_{\mathcal{D}}$ is a number of decryptions.
- Value sk had been first used by the decryption oracle $\mathcal{D}$ and then returned as a result of the 2P protocol in the first step of the protocol MLHL. This event occurs with probability $\frac{p(\kappa)\cdot n\cdot q_{\mathcal{D}}}{2^{l_k}}$, where $p(\cdot)$ is a polynomial and $q_{\mathcal{D}}$ denotes number of decryptions ($p(\kappa)\cdot n$ is number of 2P's executions).

Hence,

$$|\Pr[Succ_0'] - \Pr[Succ_0]| \leq \frac{(q_{\mathcal{E}}+q_{\mathcal{D}})^2}{2|G|} + \frac{p(\kappa)\cdot n\cdot q_{\mathcal{D}}}{2^{l_k}}.$$

Next, we simulate gradual replacement of values sk_i by random keys in the games G_1^i. We alter the simulation of Execute and Send_2^1 queries as follows: session key sk_i established during the 2P protocol between the participant P_i and server S is replaced by a random string sk_i', while we keep these randomly chosen values in a list $\Lambda_{2\mathrm{P}}$ in the format (sid_i^j,i,sk_i'). The randomly chosen values sk_i' should not repeat for any participant and any session, if some sk_i' is repeated, we stop the simulation and we let the adversary win (this happens with probability $\frac{p(\kappa)^2}{2^{l_k+1}}$, where $p(\kappa)$ specifies number of simulations of the MLHL protocol.

Game G_1^1:

In this game the session key established during the 2P protocol between participant P_1 and server S is replaced by a random string sk_1'. We store values $(sid_1^j,1,sk_1')$ in the list $\Lambda_{2\mathrm{P}}$. We show that

$$|\Pr[Succ_1^1] - \Pr[Succ_0']| \leq p(\kappa)\mathbf{Adv}^{\mathrm{AKE}}_{2\mathrm{P},\mathcal{A}_{2\mathrm{P}}}(t_{2\mathrm{P}},q_{execute},q_{send},q_{reveal}) + \frac{p(\kappa)^2}{2^{l_k+1}},$$

where $q_{send},q_{execute},q_{reveal}$ is number of Send, Execute, Reveal queries of 2P on his oracles and $p(\cdot)$ is polynomial.

To show this inequality we use a hybrid argument: we assume that there is a polynomial time distinguisher $\mathcal{A}_D$ that can distinguish games G'_0 and G^1_1 with probability $\varepsilon = |\Pr[\mathcal{A}_D^{G'_0} \to 1] - \Pr[\mathcal{A}_D^{G^1_1} \to 1]|$. We show that if ε is not negligible, we can construct an adversary $\mathcal{A}_{2\mathrm{P}}$ against the AKE security of the 2P protocol, which probability of success is not negligible. We define sequence distributions H^i_1, $i = 0 \dots p(\kappa)$. In the distribution H^i_1 the first i session keys established during the 2P protocol between participant P_1 and server S are replaced by a random string sk'_1. Clearly the distribution H^0_1 is equal to the game G'_0 and $H^{p(\kappa)}_1$ to G^1_1.

Adversary $\mathcal{A}_{2\mathrm{P}}$

1. $\mathcal{A}_{2\mathrm{P}}$ selects an index j at random from $\{1 \dots p(\kappa) - 1\}$ and a bit $b \xleftarrow{\$} \{0, 1\}$. Then $\mathcal{A}_{2\mathrm{P}}$ runs distinguisher $\mathcal{A}_D$ and responds to his oracle queries (described later). We assume that $\mathcal{A}_{2\mathrm{P}}$ is able to identify, which queries asked by $\mathcal{A}_D$ belong to the 2P protocol (Send$^1_1, \dots,$ Send$^l_1$) and which belong to the rest of the protocol MLHL (Send$^1_2, \dots,$ Send$_4$). $\mathcal{A}_{2\mathrm{P}}$ will simulate oracle queries of $\mathcal{A}_D$ as follows:
   - Send(Π^l_1, Msg) in 2P, $l < j$: $\mathcal{A}_{2\mathrm{P}}$ replies with the response of his Send(Π^l_1, Msg) oracle. If this query leads to establishment of a session key in 2P, then $\mathcal{A}_{2\mathrm{P}}$ selects a random key sk'_1 and uses it as a session key sk_1 between P_1 and S in the session sid^l_1.
   - Send(Π^j_1, Msg) in 2P: $\mathcal{A}_{2\mathrm{P}}$ replies with the response of his Send(Π^j_1, Msg) oracle. If this query leads to establishment of a session key in 2P, then $\mathcal{A}_{2\mathrm{P}}$ asks Test(Π^j_1) query and a result is used as a session key sk_1 between P_1 and S in the MLHL protocol with the session identifier sid^j_1.
   - Send(Π^l_i, Msg) in 2P, $i \neq 1 \wedge l \in \{1, \cdots p(\kappa)\}$ or $i = 1 \wedge l > j$: $\mathcal{A}_{2\mathrm{P}}$ replies with the response of his Send(Π^j_i, Msg) oracle. If this query leads to establishment of a session key in 2P, then $\mathcal{A}_{2\mathrm{P}}$ asks Reveal(Π^l_i) query and the returned result is used as a session key sk_i between P_i and S in the session sid^j_i.
   - Send(Ψ^s, Msg) in 2P: similar as Send(Π^j_i, Msg)
   - Send(Π^j_i, Msg) query outside 2P: $\mathcal{A}_{2\mathrm{P}}$ answers with the result of simulation of sending the message Msg in MLHL, while he follows rules and steps of MLHL as in the previous game. During the simulation he uses keys $sk_i, i \in \{1, 2, ..., n\}$ (which were obtained as a response of his Reveal or Test oracle or by a random choice).
   - Send(Ψ^s, Msg) query outside 2P: similar to Send(Π^j_i, Msg) outside 2P.
   - Execute$(P_1, P_2, \dots, P_n, S)$: similar to combination of the Send queries.
   - Reveal(Π^j_i): $\mathcal{A}_{2\mathrm{P}}$ answers under the rules of Reveal query in the security model (he returns a real session key sk, if Π^j_i has the key established during the simulation)
   - Test(Π^j_i): if a randomly chosen bit $b = 0$, $\mathcal{A}_{2\mathrm{P}}$ returns the real session key sk (computed during the simulation of Send or Execute queries), otherwise he returns a randomly chosen key sk'.

2. $\mathcal{A}_{2\mathrm{P}}$ returns $b \leftarrow \mathcal{D}$

We analyze the behaviour of $\mathcal{A}_{2\mathrm{P}}$ now. Fix polynomial $p(\cdot)$ and $\mathcal{A}_{2\mathrm{P}}$ chooses $j = J$, where J is a random value uniformly chosen from $\{1, \ldots, p(\kappa)\}$. If $\mathcal{A}_{2\mathrm{P}}$ gets a real session key during $\mathrm{GameAKE}_{2\mathrm{P},\mathcal{A}_{2\mathrm{P}}}$, established during the protocol 2P between participants P_1 and S, then the view of the distinguisher $\mathcal{A}_D$ is as in the distribution H_1^{J-1}. That is,

$$\Pr_{sk_1 \leftarrow 2\mathrm{P}(P_1,S)}[\mathcal{A}_{2\mathrm{P}}(sk_1) = 1 | j = J] = \Pr_{view \leftarrow H_1^{J-1}}[\mathcal{A}_D(view) = 1].$$

Since the value of j is chosen uniformly at random, we have

$$\begin{aligned}\Pr_{sk_1 \leftarrow 2\mathrm{P}(P_1,S)}[\mathcal{A}_{2\mathrm{P}}(sk_1) = 1] &= \frac{1}{p(\kappa)} \sum_{J=1}^{p(\kappa)} \Pr_{sk_1 \leftarrow 2\mathrm{P}(P_1,S)}[\mathcal{A}_{2\mathrm{P}}(sk_1) = 1 | j = J] \\ &= \frac{1}{p(\kappa)} \sum_{J=1}^{p(\kappa)} \Pr_{view \leftarrow H_1^{J-1}}[\mathcal{A}_D(view) = 1].\end{aligned}$$

If $\mathcal{A}_{2\mathrm{P}}$ chooses $j = J$ and during $\mathrm{GameAKE}_{2\mathrm{P},\mathcal{A}_{2\mathrm{P}}}$ it receives a randomly chosen value instead of the session key as a response of its Test oracle, then the view of the distinguisher $\mathcal{A}_D$ is as in the distribution H_1^J. That is,

$$\Pr_{sk'_1 \leftarrow \{0,1\}^{l_k}}[\mathcal{A}_{2\mathrm{P}}(sk'_1) = 1 | j = J] = \Pr_{view \leftarrow H_1^{J}}[\mathcal{A}_D(view) = 1].$$

Then, we have

$$\begin{aligned}\Pr_{sk'_1 \leftarrow \{0,1\}^{l_k}}[\mathcal{A}_{2\mathrm{P}}(sk'_1) = 1] &= \frac{1}{p(\kappa)} \sum_{J=1}^{p(\kappa)} \Pr_{sk'_1 \leftarrow \{0,1\}^{l_k}}[\mathcal{A}_{2\mathrm{P}}(sk'_1) = 1 | j = J] \\ &= \frac{1}{p(\kappa)} \sum_{J=1}^{p(\kappa)} \Pr_{view \leftarrow H_1^{J}}[\mathcal{A}_D(view) = 1].\end{aligned}$$

In the end we have

$$\begin{aligned}&\Big| \Pr_{sk'_1 \leftarrow \{0,1\}^{l_k}}[\mathcal{A}_{2\mathrm{P}}(sk'_1) = 1] - \Pr_{sk_1 \leftarrow 2\mathrm{P}(P_1,S)}[\mathcal{A}_{2\mathrm{P}}(sk_1) = 1] \Big| \\ &= \frac{1}{p(\kappa)} \Big| \sum_{J=1}^{p(\kappa)} \Pr_{view \leftarrow H_1^{J}}[\mathcal{A}_D(view) = 1] - \sum_{J=0}^{p(\kappa)-1} \Pr_{view \leftarrow H_1^{J}}[\mathcal{A}_D(view) = 1] \Big| \\ &= \frac{1}{p(\kappa)} \Big| \Pr_{view \leftarrow H_1^{p(\kappa)}}[\mathcal{A}_D(view) = 1] - \Pr_{view \leftarrow H_1^{0}}[\mathcal{A}_D(view) = 1] \Big| = \frac{\varepsilon}{p(\kappa)}.\end{aligned}$$

Since 2P is AKE secure protocol and $\mathcal{A}_{2\mathrm{P}}$ runs in polynomial time and $p(\cdot)$ is a polynomial, the value ε must be negligible.

$$\begin{aligned}|\Pr[Succ_1^1] - \Pr[Succ_0'']| &= |\Pr_{view \leftarrow H_1^{p(\kappa)}}[\mathcal{A}_D(view) = 1] - \Pr_{view \leftarrow H_1^{0}}[\mathcal{A}_D(view) = 1]| \\ &\leq p(\kappa)\mathbf{Adv}^{\mathrm{AKE}}_{2\mathrm{P},\mathcal{A}_{2\mathrm{P}}}(t_{2\mathrm{P}}, q_{execute}, q_{send}, q_{reveal}) + \frac{p(\kappa)^2}{2^{l_k+1}}.\end{aligned}$$

Games $G_1^2, \ldots, G_1^n$ are defined similarly. The similar reasoning of existence of a distinguisher between games G_1^i and G_1^{i+1} works. When we sum all inequalities on the left side and on the right side,

$$|\Pr[Succ_1^n] - \Pr[Succ_0']| \leq n \cdot p(\kappa)\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{2P},\mathcal{A}_{\mathrm{2P}}}(t_{\mathrm{2P}}, q_{execute}, q_{send}, q_{reveal}) + \frac{np(\kappa)^2}{2^{l_k+1}}.$$

In this part we simulate gradual replacement of values K_i by random values in the games $G_2^i, i = 1 \ldots n$. We alter the simulation of Execute queries as follows: a Diffie-Hellman value K_i established during the MLHL protocol between participants P_i and P_{i+1} is replaced by a random value K_i' from G.

Game G_2^1: We simulate everything like in the previous game in this game, however the value K_1 is replaced by a random value during Execute queries. We show that

$$|\Pr[Succ_2^1] - \Pr[Succ_1^n]| \leq p(\kappa)\mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}).$$

To prove this inequality, suppose that there exist a distinguisher $\mathcal{A}_D$ which can distinguish these two games. We can use this distinguisher to construct an adversary $\mathcal{A}_{\mathrm{DDH}}$, which can solve DDH problem, with use of similar hybrid argument as in previous games: we define a distribution H_2^i, $i \in \{0, 1, \ldots p(k)\}$. In the distribution H_2^i the values K_1 for instances Π_1^j, $j \leq i$ are chosen randomly and the values K_1 for instances Π_1^j, $j > i$ are computed as in the previous game. We assume that distinguisher $\mathcal{A}_D$ constructs $p(\kappa)$ sessions for some polynomial $p(\cdot)$ during simulation.

Adversary $\mathcal{A}_{\mathrm{DDH}}(u, v, w)$

1. $\mathcal{A}_{\mathrm{DDH}}$ chooses a random bit b and an index j.
2. $\mathcal{A}_{\mathrm{DDH}}$ answers oracle queries of the distinguisher $\mathcal{A}_D$ as follows:
   - Send, Reveal and Test queries are answered as in the previous game, Test queries are answered with the use of the bit b (note, that the adversary knows established session keys, because he simulated the execution).
   - Execute$(P_1, \ldots, P_n)$ queries are simulated in the following way:
     If instance of P_1 has form Π_1^l, where $l = j$ then simulation of Execute query for instances of participants $P_3, \ldots, P_n$ in the same session does not change. The protocol 2P between P_1, P_2 and S is simulated as in the previous game, after this simulation $\mathcal{A}_{\mathrm{DDH}}$ knows values sk_1, sk_2 – he has chosen them randomly. Then he simulates that P_1 sends a message $(ID(P_1)||\mathcal{E}_{sk_1}(u))$ and P_2 sends a message $(ID(P_2)||\mathcal{E}_{sk_2}(v))$ then K_1 is set to w, $K_2 = v^{x_3}$, $K_n = u^{x_n}$. Next he continues with the simulation of the rest of MLHL. Other Execute queries, where $l \neq j$ are simulated as follows:
     - If instance of P_1 has form Π_1^l, where $l < j$ then $\mathcal{A}_{\mathrm{DDH}}$ starts to simulate 2P between participants $P_1, \ldots, P_n$ and S as in the previous game. After simulation of the 2P protocol he chooses randomly keys sk_i', $i = 1 \ldots n$ and simulates the rest of the MLHL as in the previous game however, a computed value K_1 in each session is replaced by a random value.

– If instance of P_1 has form Π_1^l, where $l > j$ then $\mathcal{A}_{\mathrm{DDH}}$ starts to simulate 2P between participants $P_1, \ldots, P_n$ and S as in the previous game. After simulation of the 2P protocol he continues with simulation of the rest of the MLHL as in the previous game.

3. $\mathcal{A}_{DDH}$ returns a $b \leftarrow \mathcal{A}_D$

If (u, v, w) from adversary's input is a DDH triple and the index $j = 0$, the view of the distinguisher $\mathcal{A}_D$ is the same as in the game G_1^n (H_2^0). If (u, v, w) is not a DDH triple and the index $j = p(k)$, the view of $\mathcal{A}_D$ is the same as in the game G_2^1 ($H_2^{p(k)}$). Thus the advantage of $\mathcal{A}_{\mathrm{DDH}}$ is at least as great as $\frac{1}{p(\kappa)}$ of the advantage of $\mathcal{A}_D$ (we skip the detailed reasoning).

$$|\Pr[Succ_2^1] - \Pr[Succ_1^n]| = p(\kappa)\mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH})$$

The games $G_2^2, \ldots, G_2^n$ are defined similarly. When we sum inequalities, we have

$$|\Pr[Succ_2^n] - \Pr[Succ_1^n]| = n \cdot p(\kappa) \cdot \mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}).$$

Game G_3:
In this game the session key of MLHL is replaced by a random value during Execute queries. We have

$$\Pr[Succ_3] = \Pr[Succ_2^n].$$

This claim follows from the view of an adversary in this two games. In the game G_2^n the values K_i are chosen at random, therefore they are independent from previously sent messages (They are not sent directly, but as xor-ed values w_i, which can originate from combination of $2^{|w_i|}$ different pairs of values). This implies that the computed w_i (which adversary can see) are independent from previously sent messages. From all of this facts follows, that the computed session key is independent from all sent values and therefore there is no difference between these games.
Game G_4:
In this game we change simulation of the first subcase of the decryption oracle (defined in the game G_0') in Send queries: we build an instance of PDDH problem in simulation of the protocol. We set $\beta = 0$, thus the challenger $Chall^\beta(\cdot)$ returns vectors $(\zeta_1, ..., \zeta_n, \gamma_1, ..., \gamma_n)$ from the distribution PDH_n^*. New vectors are returned in every session, however the same vectors are returned in queries on the same session. For randomly chosen $(\alpha_1, ..., \alpha_n)$, $\alpha_i \xleftarrow{\$} Z_q^*$, vectors $(\zeta_1^{\alpha_1}, ..., \zeta_n^{\alpha_n}, \gamma_1^{\alpha_1\alpha_2}, ..., \gamma_n^{\alpha_n\alpha_1})$ have equal distribution to the original $(\zeta_1, ..., \zeta_n, \gamma_1, ..., \gamma_n)$. We use this property for application of random self-reducibility of the PDDH problem. The decryption is changed as follows:

- $\mathcal{D}_{sk_i}(z^*)$ – if $(sid_i^j, i, sk_i) \in \Lambda_{2\mathrm{P}}$, $(\zeta_1, ..., \zeta_n, \gamma_1, ..., \gamma_n) \leftarrow Chall^\beta(sk_1, ..., sk_n)$ (the arguments of $Chall^\beta$ can be found in the $\Lambda_{2\mathrm{P}}$ list sharing the same value of the session ID), we choose $\alpha_i \xleftarrow{\$} Z_q^*$ randomly and compute $z_i = \zeta_i^{\alpha_i}$. If $(\cdot, \cdot, \cdot, \cdot, sk_i, z_i, \cdot) \in \Lambda_{\mathcal{E}}$, then we stop the simulation, adversary wins. Otherwise we add record $(dec, sid_i^j, i, \alpha_i, sk_i, z_i, z^*)$ to $\Lambda_{\mathcal{E}}$ and return z_i.

Exponent α_i specifies, how we applied random self-reducibility of PDDH problem on instance generated by the challenger. Exponent α_i can be defined in the list $\Lambda_{\mathcal{E}}$ only if values sid_i^j and i are known. The view of the adversary does not change and therefore we have

$$\Pr[Succ_4] = \Pr[Succ_3].$$

Game G_5:
We change the simulation of Send_2^1, Send_2^2 and Send_3 queries. First, encryption of messages in the second step of the protocol is changed during simulation of Send_2^1. The instance Π_i^j chooses $z_i^* \xleftarrow{\$} G$ randomly and computes $z_i = \mathcal{D}_{sk_i}(z_i^*)$ as in the previous game. Then Π_i^j sends a message $(ID(P_i)||z_i^*)$. Therefore Send_2^1 queries in the second step of the MLHL lead to adding of α_i to the list $\Lambda_{\mathcal{E}}$. Simulation ends if

- $(enc, \perp, \perp, \perp, sk_i, \cdot, z_i^*) \in \Lambda_{\mathcal{E}}$, because we do not know the value of α_i. This possibility occurs if the adversary asks for encryption of some value with the key sk_i and the result of encryption was z_i^* (it means that $(enc, \perp, \perp, \perp, sk_i, \cdot, z_i^*) \in \Lambda_{\mathcal{E}}$). The probability of this event is $q_{\mathcal{E}}/2^{l_k}$. In this case we stop the simulation, the adversary wins.
- $(dec, \perp, \perp, \perp, sk_i, z_i, z_i^*) \in \Lambda_{\mathcal{E}}$. This possibility occurs if we decrypt the value z_i^*, while the values i, sid_i^j belonging to sk_i were not known. However this situation can not occur (see the Game G_0', point 3).

When the server accepts the message $(ID(P_i)||z_i^*)$ during simulation of Send_2^2, he should resend it to participants P_{i-1} and P_{i+1}, thus he must decrypt z_i^*. The following cases can occur:

- z_i^* was encrypted in the aforementioned manner, thus we know the value α_i. We can continue with the simulation of encryption described bellow.
- z_i^* is response of the encryption oracle $\mathcal{E}_{sk_i}$, while sk_i is a correct key of Π_i^j in the corresponding session (thus adversary guessed the sk_i and used it for encryption of data for server). In this case we stop the simulation, adversary wins. This event occurs with probability $q_{\mathcal{E}}/2^{l_k}$.
- z_i^* was chosen by the adversary without asking the encryption oracle. In this situation the adversary does not know the password and therefore he could not compute messages in the way they go through the control step. The simulation continues as follows: we compute $z_i' = \mathcal{D}_{sk_i}(z_i^*)$, then we compute $z_i^{**} = \mathcal{E}_{sk_{i+1}}(z_i')$ and send to the user P_{i+1} a message $(ID(S)||ID(P_i)||z_i^{**})$. Similar for P_{i-1}. Next we continue in simulation as in previous games, however, the adversary does not know any of values K_i, therefore he could not manipulate other messages in the way they go through the verification step of the MAC scheme, unless he breaks it with probability $\mathbf{Adv}_{\text{M},\mathcal{A}_\text{M}}^{\text{MAC}-\text{forge}}(t')$, which is negligible.

Encryption of $z_i = \mathcal{D}_{sk_i}$ (in the first case) with another passwords (sk_{i-1} and sk_{i+1}) (in second step) works as follows:

- $\mathcal{E}_{sk_{i+1}}(z_i)$
  - if $(\cdot,\cdot,\cdot,\cdot,sk_{i+1},z_i,\cdot) \notin \Lambda_{\mathcal{E}}$ and $(dec, sid_i^j, i, \alpha_i, sk_i, z_i, \cdot) \in \Lambda_{\mathcal{E}}$ (this record was added in the simulation described above by the instance Π_i^j), then we choose $z^{**} \xleftarrow{\$} G$, if $(\cdot,\cdot,\cdot,\cdot,sk_{i+1},\cdot,z^{**}) \notin \Lambda_{\mathcal{E}}$, we return z^{**} and add record $(enc, sid_i^j, i, \perp, sk_{i+1}, z, z^{**})$ into $\Lambda_{\mathcal{E}}$, else we stop the simulation and the adversary wins.
  - if $(enc, \perp, \perp, \perp, sk_{i+1}, z_i, \cdot) \in \Lambda_{\mathcal{E}}$, we stop the simulation and the adversary wins. This case occurs if the adversary asked for encryption of value z_i with key sk_{i+1}. The probability of this event is $q_{\mathcal{E}}/2^{l_k}$.
  - if $(enc, sid_i^j, i+2, \perp, sk_{i+1}, z_i, z^*) \in \Lambda_{\mathcal{E}}$, we return z^*. This case occurs if during the simulation of execution of the protocol a request for resending the value z_i to the instance Π_{i+1}^j (sent with the instance Π_{i+2}^j) happens, while the value was encrypted with the key sk_{i+1}.
  - if $(dec, sid_i^j, i+1, \alpha_{i+1}, sk_{i+1}, z_i, z^*) \in \Lambda_{\mathcal{E}}$, we return z^*. This case can occur by simulation of Send queries of the instance Π_{i+1}^j in the second round.
- $\mathcal{E}_{sk_{i-1}}(z_i)$ – similar to the previous case.

The simulation of Send$_3$, when Π_i^j receives a messages $(ID(S)||ID(P_{i-1})||z_{i-1}^*)$ and $(ID(S)||ID(P_{i+1})||z_{i+1}^*)$ works as follows: compute $z_{i-1} = \mathcal{D}_{sk_i}(z_{i-1}^*)$ and $z_{i+1} = \mathcal{D}_{sk_i}(z_{i+1}^*)$. This three cases can occur:

- z_{i-1}^* and z_{i+1}^* were encrypted in the previous manner. We can continue with the simulation as described bellow.
- one or both z_{i-1}^* and z_{i+1}^* is/are the answer from query on the encryption oracle $\mathcal{E}_{sk_i}$, while sk_i is correct key of the instance Π_i^j in the session j (thus adversary guessed a password and used its for encryption of data from server). This event occurs with probability $q_{\mathcal{E}}/2^{l_k}$. In this case we stop the simulation, adversary wins.
- one or both z_{i-1}^* and z_{i+1}^* was/were chosen by the adversary without asking for the Encryption oracle, in this situation the adversary does not know the password and therefore he could not compute messages in the way they go through the verification step of the MAC scheme, unless he breaks it with probability $\mathbf{Adv}_{\mathrm{M},\mathcal{A}_{\mathrm{M}}}^{\mathrm{MAC-forge}}(t_M)$, which is negligible.

If messages were sent as we simulate them, we have $z_i = \zeta_i^{\alpha_i}, z_{i-1} = \zeta_{i-1}^{\alpha_{i-1}}, z_{i+1} = \zeta_{i+1}^{\alpha_{i+1}}$ and we can compute

$$K_{i-1} = \mathcal{H}(CDH(z_{i-1}, z_i)), K_i = \mathcal{H}(CDH(z_i, z_{i+1}))$$

$$w_i = K_{i-1} \oplus K_i, \tau_i = \mathrm{Mac}_{K_i}(w_i)$$

and resend a message $(ID(P_i), w_i, \tau_i)$. When every participant broadcasts such message, the session key can be computed.

This game is the same as the previous unless mentioned "bad" events happen.

$$|\Pr[Succ_5] - \Pr[Succ_4]| \leq 4q_{\mathcal{E}}/2^{l_k} + 2\mathbf{Adv}^{\mathrm{MAC-forge}}_{\mathrm{M},\mathcal{A}_{\mathrm{M}}}(t_M)$$

Game G_6:
In this game we change the bit β to 1, thus the values $(\zeta_1, ..., \zeta_n, \gamma_1, ..., \gamma_n)$ are from distribution $PDH^{\$}_n$. Clearly holds that

$$|\Pr[Succ_6] - \Pr[Succ_5]| \leq p(\kappa) \cdot \mathbf{Adv}^{\mathrm{PDDH}_n}_{G,\mathcal{A}_{\mathrm{PDDH}}}(t_{DDH}),$$

where $p(\kappa)$ is the number of sessions, p is a polynomial.
Game G_7:
The session key of MLHL is replaced by a random value during Send queries in this game. We have

$$\Pr[Succ_7] = \Pr[Succ_6].$$

This claim follows the view of the adversary in this two games. In the game G_6 values K_i are chosen randomly, therefore they are independent from previous sent messages (They are not sent directly, but as xor-ed values w_i, which can originate from combination of $2^{|w_i|}$ different pairs of values). This implies that the computed w_i (which adversary sees) are independent from previous sent messages. From all of this facts follows, that the computed session key is independent from all sent values and therefore there is no difference between these games.
The probability of the adversary's success in this game is $\Pr[Succ_7] = \frac{1}{2}$, because the session key is randomly chosen and independent from the previous messages. When we sum all (in)equalities of games, we have:

$$\begin{aligned}|\Pr[Succ_7] - \Pr[Succ_0]| \leq{}& \frac{(q_{\mathcal{E}} + q_{\mathcal{D}})^2}{2|G|} + \frac{p(\kappa) \cdot n \cdot q_{\mathcal{D}}}{2^{l_k}} + 2\mathbf{Adv}^{\mathrm{MAC-forge}}_{\mathrm{M},\mathcal{A}_{\mathrm{M}}}(t_M) \\ &+ p(\kappa) \cdot n\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{2P},\mathcal{A}_{\mathrm{2P}}}(t_{\mathrm{2P}}, q_{execute}, q_{send}, q_{reveal}) \\ &+ \frac{np(\kappa)^2}{2^{l_k+1}} + p(\kappa) \cdot n \cdot \mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}) + 4q_{\mathcal{E}}/2^{l-k} \\ &+ p(\kappa) \cdot \mathbf{Adv}^{\mathrm{PDDH}_n}_{G,\mathcal{A}_{\mathrm{PDDH}}}(t_{DDH})\end{aligned}$$

$$\begin{aligned}\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{MLHL},\mathcal{A}_{\mathrm{MLHL}}}(\mathcal{A}) \leq 2\Bigg(& \frac{(q_{\mathcal{E}} + q_{\mathcal{D}})^2}{2|G|} + \frac{p(\kappa) \cdot n \cdot q_{\mathcal{D}}}{2^{l_k}} + 2\mathbf{Adv}^{\mathrm{MAC-forge}}_{\mathrm{M},\mathcal{A}_{\mathrm{M}}}(t_M) \\ &+ p(\kappa) \cdot n\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{2P},\mathcal{A}_{\mathrm{2P}}}(t_{\mathrm{2P}}, q_{execute}, q_{send}, q_{reveal}) \\ &+ \frac{np(\kappa)^2}{2^{l_k+1}} + 2p(\kappa) \cdot n \cdot \mathbf{Adv}^{\mathrm{DDH}}_{G,\mathcal{A}_{\mathrm{DDH}}}(t_{DDH}) + 4q_{\mathcal{E}}/2^{l-k}\Bigg).\end{aligned}$$

□

Privacy-Preserving Verification of Clinical Research

Eleftheria Makri[1] Maarten H. Everts[1,3] Sebastiaan de Hoogh[2]
Andreas Peter[1] Harm op den Akker[1,4] Pieter H. Hartel[1,3]
Willem Jonker[1]

[1] University of Twente
[2] Eindhoven University of Technology
[3] TNO, Netherlands Organisation for Applied Scientific Research
[4] Roessing Research and Development

Abstract: We treat the problem of privacy-preserving statistics verification in clinical research. We show that given aggregated results from statistical calculations, we can verify their correctness efficiently, without revealing any of the private inputs used for the calculation. Our construction is based on the primitive of Secure Multi-Party Computation from Shamir's Secret Sharing. Basically, our setting involves three parties: a *hospital*, which owns the private inputs, a *clinical researcher*, who lawfully processes the sensitive data to produce an aggregated statistical result, and a third party (usually several *verifiers*) assigned to verify this result for reliability and transparency reasons. Our solution guarantees that these verifiers only learn about the aggregated results (and what can be inferred from those about the underlying private data) and nothing more. By taking advantage of the particular scenario at hand (where certain intermediate results, e.g., the mean over the dataset, are available in the clear) and utilizing secret sharing primitives, our approach turns out to be practically efficient, which we underpin by performing several experiments on *real* patient data. Our results show that the privacy-preserving verification of the most commonly used statistical operations in clinical research presents itself as an important use case, where the concept of secure multi-party computation becomes employable in practice.

1 Introduction

Statistical analysis of experimental data is the cornerstone in many research areas. However, human error and fraud are common threats to the integrity of the statistical results [Fan09, Ens, Mis, RBG+00]. In addition, verification of such statistical results cannot be applied in a straight-forward manner, since in many cases the underlying data has to remain confidential. To address this problem, we propose a privacy-preserving verification procedure that allows a number of semi-honest verifiers to ascertain that statistical calculations are consistent with the confidential data that they are supposedly based on, without learning about this underlying confidential data.

In medical research, it is common practice to give clinical researchers access to raw patient data. This is necessary for researchers to determine the appropriate statistical analysis

method for the specific dataset in question [Fie09, pp. 822]. Patient privacy in that context is preserved by the researchers themselves, who are bound by confidentiality agreements. Currently, only the most prestigious medical scientific journals like Thorax [Tho] perform statistics verification on the clinical research results, prior to their publication. This is a labor intensive task, which has to be performed by expert statisticians. In addition, there is a trade-off between patient privacy and thoroughness of the verification procedure. On the one hand, if the results are only partially checked, thoroughness is sacrificed to preserve some privacy. On the other hand, if all the results are thoroughly checked, then patient privacy will be completely compromised. Note that current anonymization techniques have been shown insecure, since anonymized data can be de-anonymized [Swe02]. Thus, disclosure of (possibly anonymized) patient information should not be allowed; not even to medical journals for verification.

Although hospitals have the confidential patient data used for the statistical analysis available in the clear, they currently do not consider the verification of the statistics. This is because 1) hospitals wish to avoid the additional workload brought by the verification; 2) clinical researchers are usually employed by the same hospital that provides them with the data, where a conflict of interests might arise; 3) on-site verification does not scale, since it is not possible to verify the results accruing from datasets of different hospitals (i.e., in the case of a multi-center clinical research). In contrast, medical journals are interested in the correctness of the results that they publish. Therefore, we propose that journals outsource the verification of statistics to an independent group of servers, called the *verifiers*, in a privacy-preserving manner. In this setting, the architecture that we propose is depicted in Fig. 1. Our approach can be fully automated and does not require additional manpower to be employed. This may well serve as a motivation for all (medical) journals to implement this paradigm and integrate verification into their pre-publication process.

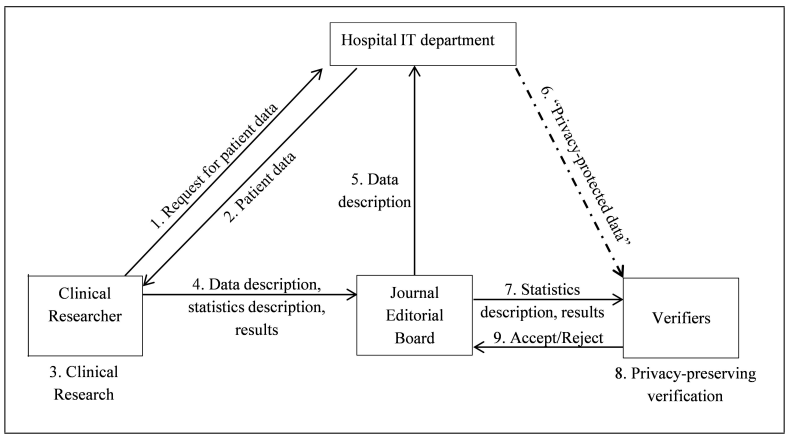


Figure 1: Privacy-preserving architecture for the verification of clinical research

Concretely, we make the following contributions:

Enhance Privacy-Awareness in the Verification of Clinical Research. Patient data is confidential and is only to be disclosed to (trusted) experts conducting the clinical research

with the explicit patients' informed consent. External parties, such as medical journals, should not receive patient confidential information, even if it has been anonymized. We point out this important issue and propose a best practice framework (Fig. 1).

Enable Privacy-Preserving Verification of Clinical Research. As the verification of clinical research at the hospital site is unsuitable (for the mentioned reasons, such as conflict of interests), we propose a mechanism that allows for the outsourcing of this verification to several semi-honest *verifiers* without compromising the confidentiality of the patients' data. Our approach is based on secure multi-party computation from Shamir's secret sharing and is proven secure in the semi-honest model. We base our protocols on secret sharing, because of the storage and computational efficiency it provides. For Shamir's secret sharing to be secure, while allowing the evaluation of polynomial functions over the shares, we need at least three computing parties. Hence, we assume several (a minimum of three) *independent*, non-colluding verifiers performing the verification task. The contractual relationship between the journal and the verifiers, prevents the verifiers from cheating. Thus, our application scenario allows us to work in the semi-honest model, where the parties are assumed to follow the protocol correctly, but they are able to record the protocol transcript, in order to infer private information.

Demonstrate the Practicality of our Approach with Real Patient Data. We develop a set of privacy-preserving algorithms, which allows the verification of the most commonly used statistical operations in clinical research [Md09, OS08, ZBT07]: mean, variance, Student's t-test, Welch's t-test, ANOVA (F-test), simple linear regression, chi-squared test, Fisher's exact test, and McNemar's test. We test this representative set of algorithms on a *real* medical dataset, produced in the context of a tele-treatment medical research [AodJH10], and show their efficiency. The dataset consists of 2370 activity feedback messages produced for 85 patients, participating in the research. We also doubled and tripled this dataset to show the scalability of our solution.

The execution times of our verification algorithms on this dataset range from 43.5 ms in the fastest case (a verification of the mean age of 84 patients) to 884.6 ms in the slowest case (a verification of simple linear regression on 6828 messages). We refer to Section 5.2 for details. Our execution times, in combination with the fact that we use real data, substantiates the practicality of our approach.

Our solution can detect any error in the computation of the statistics, but it cannot detect logical errors (i.e., errors in the selection of the appropriate statistical model that fits the data best). Thus, our solution is not meant to substitute, but rather to enhance the peer-reviewing process of medical journals. Currently, the statistics that our solution can treat, are limited to the ones meant for normally distributed data. Dealing with statistics meant for not normally distributed data requires highly efficient ranking of the inputs, which is a challenging task in the secret shared domain. We leave it as an interesting future work.

The rest of the paper is organized as follows: In Section 2, we discuss related work and frame our contribution within it. In Section 3, we give the preliminaries on the secret sharing and the multi-party computation techniques that we utilize. In Section 4, we discuss the proposed verification algorithms and the statistics to be verified. Section 5 deals with the security and performance analysis of our algorithms and Section 6 concludes the paper.

2 Related Work

A lot of work has been done to solve the problem of privacy-preserving statistical analysis [DA01, DHC04, SCD+08, DN04, HR10, DF97, KLSR09]. In this setting, aggregated statistical results are being computed among several parties, while the inputs for the calculation of these results remain private. In contrast to our work, their attention is focused on secure *computation* of certain statistics, where each party that is involved in the computation provides its own private data. We deal with the *verification* of statistics, for which the verifiers do not contribute their own private data. In that context, privacy concerns data provided by a party, which is not involved in the computation.

Another related topic in our setting is *verifiable computation* [GGP10, PRV12, AIK10, BGV11, PST13]. Verifiable computation allows a party (or set of parties) to outsource the computation of certain functions to untrusted external parties, while maintaining *verifiable* results. However, despite their lack of efficiency, all existing constructions in this area are not applicable in our setting, as the verification is not meant to be privacy-preserving. The security definitions of these works guarantee that the untrusted computing parties cannot cheat in the computation (i.e., a false result will not pass the verification procedure).

Another recently emerged approach towards addressing the problem of privacy-preserving verification is computing on authenticated data [ABC+12], which can be accomplished using homomorphic signatures. The concept of homomorphic signatures emerged from [JMSW02], where the first example application scenarios and definitions were given. Recently, the application of homomorphic signatures was extended from treating only set operations, to computation of functions on the signed data [BF11b, BF11a]. In [BF11b], Boneh and Freeman propose a homomorphic signature scheme that allows linear functions to be performed on the signed data, while anyone can produce a signature on the result of the function. The scheme is weakly context hiding, meaning that it protects the privacy of the signed data, but not hiding the fact that the function was executed. Although the aforementioned requirement fits well in our scenario, the scheme can only treat linear functions, which is insufficient for our purposes, as we need to compute additions and multiplications with non-constants. Building upon their previous work [BF11b], Boneh and Freeman [BF11a] propose a homomorphic signature scheme for polynomial functions. However, the enhanced functionality of the latter scheme [BF11a] comes at the cost of completely loosing the property of context hiding. This makes it inadequate for our scenario, as it provides no privacy of the underlying inputs at all.

Finally, Thompson et al.'s work [THH+09] deals with computing and verifying aggregate queries on (private) outsourced databases. They look at a setting, where a data owner outsources his private database to third-party service providers, who can later answer aggregate queries of external users or the data owner himself. Although their work is based on similar building blocks as ours, their setting is slightly different as it focuses on the computation of a very limited class of statistics and not on their verification. Exploiting the fact that we only perform verification (instead of computation from scratch) allows us to achieve very high efficiency compared to their work, while being able to deal with more sophisticated statistics at the same time.

The efficiency and hence practicality of Multi-Party Computation has received a lot of attention in the last few years [BCD+09, BSMD10, BLW08]. Our solution, similarly to the work of Bogetoft et al. [BCD+09] is based on VIFF [VD], which we used to implement the proposed privacy-preserving verification algorithms.

3 Preliminaries

We recall Shamir's secret sharing [Sha79] and Secure Multi-Party Computation [GRR98], which form the main building blocks of our protocols.

3.1 Shamir's Secret Sharing

A secret sharing scheme is a protocol, where a special party called the *dealer*, wishes to share a secret value $s \in \mathbb{F}_q$ (i.e., a secret value in a finite field of order q), among the set of protocol participants. A subset of the protocol participants, called the *qualified set*, can reconstruct the original secret value. Shamir's secret sharing scheme [Sha79] consists of three subprotocols: the share generation, secret sharing and secret reconstruction. During the share generation, the *dealer* of the protocol chooses a random polynomial (over some finite field $\mathbb{F}_q$) $p(x) = \alpha_t x^\tau + ... + \alpha_1 x + s$ of degree τ, where $\tau+1$ is the number of players in the qualified set and s is the secret to be shared. Then, he evaluates the polynomial for each player as follows: $[s]_i := p(i)$, where $[s]_i$ denotes the share of s of the i^{th} player P_i. For secret sharing all the shares have to be distributed to the players via secure channels. Then, to reconstruct the secret, anyone who possesses at least $\tau + 1$ shares can interpolate the polynomial (e.g., by Lagrange interpolation) and reconstruct the original secret s.

3.2 Secure Multi-Party Computation based on Secret Sharing

Due to the additively homomorphic property of Shamir's secret sharing scheme, addition and subtraction of Shamir's shares can be performed directly on the shares (locally by each player), without requiring any interaction. This means that after having added/subtracted the shares, we can reconstruct the resulting secret, and the result of the reconstruction will be the correct result of the summation/subtraction. The same holds for addition, subtraction and multiplication with public constants on Shamir's shares. Computing the sum of the shares of a secret vector's elements and then opening this result is commonly used for the construction of our privacy-preserving verification algorithms; we will denote the protocol for the aforementioned computation SumPub(.). The input for this protocol is the shares of a secret shared vector of integer values, and the output is the result of the addition of the vector elements (in the clear).

To multiply with shares, we need an interactive protocol to be executed among the players. In [GRR98], an interactive protocol for multiplication of Shamir's shares is proposed,

which completes its execution in one communication round. This protocol reduces the degree of the underlying polynomial (which has been doubled due to the multiplication) and restores its randomness, requiring one interactive operation for both. Our verification algorithms heavily depend on the computation of inner products in the secret shared domain. For the computation of inner products, we use a generalized version of the multiplication protocol [GRR98], also discussed in [CHd10], that comes at the same round communication cost as the multiplication (i.e., one communication round). This protocol (called InnerPub(.)) performs all the necessary operations in the secret shared domain and then reconstructs the result of the inner product. This is done by first using the PRZS(.) protocol (Pseudorandom Zero-Sharing), proposed in [CDI05], to distribute the shares of a 0 to the protocol participants; those shares are later added to the share of the multiplication result, to restore its randomness. The summation of the products and the reconstruction of the result together, require only one communication round, in which the shares are distributed, the summation is computed and the result is reconstructed. The input for the InnerPub(.) protocol is the shares of two secret shared vectors of integer values, and the output is the result of computing the inner product on these vectors (in the clear).

4 Privacy-Preserving Statistics Verification

In this section, we describe our protocols for privacy-preserving verification of statistics, while dealing with each statistical test separately. Prior to the execution of each protocol, we assume that the original data used in the calculation has been properly secret shared among the verifiers (by the hospital), according to Shamir's Secret Sharing scheme [Sha79]. The main ingredients for our verification protocols, which act in the secret shared domain, are the InnerPub(.) and SumPub(.) protocols discussed in the Preliminaries.

Our approach takes advantage of the fact that all statistical results to be verified are public. Thus, we can let the verifiers recompute the statistics in the secret shared domain, reconstruct the shared results, and compare with the statistics to be verified in the clear. In most cases, we can also take advantage of intermediate results, since they are public. Note that although our building blocks work on integers, we can also handle fixed point numbers, by scaling them up to the desired precision and then treat them as integers.

4.1 Privacy-Preserving Mean Verification

The mean (or average) is one of the simplest statistics and is computed as

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N}, \tag{1}$$

where N is the size of the sample (i.e., the number of individual subjects to be analyzed) and x_i is the variable concerning the i^{th} subject.

Recall that we assume that all x_i ($i = 1, .., N$) have been secret shared among the verifiers. We compute the value $\bar{x}$ in a privacy-preserving manner, by letting the verifiers run SumPub(.). This yields the result $\sum_{i=1}^{N} x_i$. The opening of this result does not constitute a privacy violation, since both the mean value to be verified and the size of the sample are public values, and the result of the summation can be computed by those two values.

By taking advantage of the fact that the number of subjects N is public, we do not perform the division by N in the secret shared domain. Instead, we multiply the mean value (that is to be verified) by N and compare it with the sum $\sum_{i=1}^{N} x_i$ that the verifiers computed (for efficiency reasons). The pseudo-code is given in Algorithm 1 (Appendix A).

4.2 Privacy-Preserving Variance Verification

The variance is a measure, assessing how far the real observations are from the expected value (i.e., the mean value discussed in the previous subsection) and is computed as

$$S^2 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}. \tag{2}$$

The variance only makes sense if published together with the mean value. Thus, the mean value corresponding to the variance will also be public. Our variance verification algorithm first verifies the aforementioned mean value, since it cannot be safely used, unless verified. Next, the verifiers compute all the N subtractions locally on the shares of x_i and then interactively compute an inner product (with public result) on the results of these subtractions. This is performed by invoking the InnerPub(.) protocol, discussed earlier. Similarly to the mean verification protocol, we avoid the division by $N-1$ in the secret shared domain and perform a multiplication of the received variance (to be verified) by $N-1$, instead. Then, the verifiers check the consistency of the latter product, with the inner product computed earlier and if any of them fails to find a match, the verification fails. Details are given in Algorithm 2 (Appendix A).

4.3 Privacy-Preserving Student's t-test Verification

The Student's t-test is one of the most frequently used statistical tests to assess the significance of a statistical hypothesis and is calculated as

$$t_{student} = \frac{\frac{\sum x_i}{N} - \frac{\sum y_i}{N}}{\sqrt{\frac{\frac{\sum (x_i - \bar{x})^2}{N-1} + \frac{\sum (y_i - \bar{y})^2}{N-1}}{N}}} = \frac{\bar{x} - \bar{y}}{\sqrt{\frac{S_x^2 + S_y^2}{N}}}, \tag{3}$$

where S_x^2 (resp. S_y^2) is the variance of variable x (resp. y).

The Student's t-test depends on: the variables (x and y) on which it is computed, which we treat as the private inputs that are secret shared among the verifiers; the mean and variances

corresponding to those variables; and the number of subjects N. In the context of clinical research, the variables x and y can be blood pressures of N patients, after being treated with medication X and Y respectively, and the hypothesis could concern the effectiveness of these two medications in reducing the blood pressure. As we will use the public means and variances for the verification of the t-value, we first need to verify them. After having verified the public mean values and the corresponding variances, the verifiers act only in the plaintext domain to evaluate equation (3), and check the consistency of this result with the one received for verification. For further details see Algorithm 3 (Appendix A).

4.4 Privacy-Preserving Welch's t-test Verification

The Welch's t-test is a variation of the Student's t-test, for the cases where the two groups x, y under consideration, have different variances and sizes. The formula for calculating Welch's t-test is given by

$$t_{welch} = \frac{\bar{x} - \bar{y}}{\sqrt{\frac{S_x^2}{N_x} + \frac{S_y^2}{N_y}}}, \tag{4}$$

where N_x (resp. N_y) is the size of group x (resp. y).

After having verified the mean values and the corresponding variances, there is no other calculation or verification to be performed in the secret shared domain. Having verified the means and variances for equation (4), and given that N_x, N_y are public, the verifiers evaluate equation (4). To complete the verification, the verifiers compare the result of the aforementioned privacy-preserving evaluation with the t-value to be verified. The details of the verification algorithm for Welch's t-test are given in Algorithm 4 (Appendix A).

4.5 Privacy-Preserving F-test Verification

The F-test is one of the most commonly used tests as part of the Analysis Of Variance (ANOVA). ANOVA is used to determine the significance of a statistical hypothesis. It is used instead of a t-test when there are more than two groups for which the significance of the difference among them needs to be determined. We can compute the F-value as

$$F = (\sum_{i=1}^{K} \frac{N_i(\bar{x}_i - \bar{X})^2}{K-1}) / (\sum_{i=1,j=1}^{K,N_i} \frac{(x_{ij} - \bar{x}_i)^2}{G-K}), \tag{5}$$

where K is the number of groups under analysis, N_i is the size of the i^{th} group, $\bar{x}_i$ is the mean of the i^{th} group, $\bar{X}$ is the mean of all group means, x_{ij} is the j^{th} (private) value of group i, and G is the total size (i.e., the sum of all N_i). The privacy-preserving variant of this, works as follows: the verifiers compute and verify the means and variances of all groups. Then, they compute in the clear the overall mean $\bar{X}$ and the total size G. Given

the aforementioned intermediate results, the verifiers evaluate equation (5) and compare the result to the F-value to be verified. The detailed protocol is listed in Algorithm 5.

4.6 Privacy-Preserving Simple Linear Regression Verification

Simple linear regression is used to predict an outcome (or dependent) variable from one predictor (or explanatory) variable. Specifically for simple linear regression, what needs to be calculated is the coefficients of the straight line

$$y = \alpha x + \beta \tag{6}$$

where the coefficient β is given as $\beta = \frac{\sum_{i=1}^{N}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{N}(x_i-\bar{x})^2}$, and α is given as $\alpha = \bar{y} - \beta\bar{x}$. The main ingredients for the calculation of β are the mean values of each of the two groups. Thus, the verifiers begin with verifying the means. Observe also that the denominator of the fraction is the sum of squared errors, which can be computed by invoking the InnerPub(.) protocol with the appropriate arguments (as seen before in the verification of the variance). The numerator of the fraction is also an inner product that the verifiers compute. The result of this inner product is allowed to be revealed to the verifiers, because the coefficient β is part of the public result of linear regression and this value can be directly determined by multiplying β with the sum of squared errors over variable x, which is also public (as part of the variance). Having calculated all the aforementioned, the verifiers compute β and compare it with the received one. Next, they calculate α and proceed to its comparison with the α received for verification. After those two comparisons, the parameters of the straight line, accruing from the simple linear regression have been verified. Our detailed protocol for this test is listed in Algorithm 6 (Appendix A).

4.7 Privacy-Preserving Chi-Squared test Verification

The Chi-squared test [Pea00] and the two tests that follow in the next subsections, namely Fisher's exact test [Fis22] and McNemar's test [McN47], are all statistical tests meant to be used when the underlying data is categorical. This means that those tests examine the frequency distributions of observations in a group. The frequencies $observed_{ij}$ are recorded in a table, called the contingency table, where the row and column totals (i.e., the sum of each row's elements $RowTotal_i$ and column's elements $ColumnTotal_j$, respectively) are also recorded. The most well-known chi-squared test (and the one that we treat in this paper) is Pearson's χ^2-test [Pea00]. In particular, this test measures how well the experimental data fits in the chi-squared distribution. The formula for Pearson's χ^2-test is

$$\chi^2 = \sum \frac{(observed_{ij} - model_{ij})^2}{model_{ij}}, \tag{7}$$

where $model_{ij} = (RowTotal_i \cdot ColumnTotal_j)/N$.

To enable privacy-preserving verification of tests meant for categorical data, a preprocessing step is required. During the preprocessing phase, the raw data is encoded to its unary representation. The number of categories, in each dimension of the clinical research, defines the number of bits of each entry in the table of raw data. For example, if we were examining the effect of 3 different medications, the number of bits of each entry in the medication column would have been also 3. After having successfully preprocessed the data, the hospital secret shares this data bitwise among the verifiers. We require this preprocessing step, as it allows us to efficiently compute the frequencies in the contingency table, by only adding up the data, or computing their inner product column-wise.

For the verification of the χ^2-test, we need to verify the frequencies of the variables in a contingency table. This is common to all statistical tests meant for categorical data and we present this step in a separate algorithm (VrfContingency(.), Algorithm 7 in Appendix A), for reusability purposes. The contingency table verification is performed by calculating the inner product of each bitwise secret shared variable's column, with the second variable's corresponding column. This is what the verifiers compute in the secret shared domain and then check its consistency with the table received for verification. The total and the marginal totals do not require calculations in the secret shared domain to get verified, since they can be computed by adding up the (verified) frequencies. Having computed the total and marginal totals, and verified the frequencies in the contingency table, the verifiers evaluate equation (7) and compare the result to the value to be verified. Our detailed algorithm for privacy-preserving χ^2-test verification is given in Algorithm 8 (Appendix A).

4.8 Privacy-Preserving Fisher's exact test Verification

Fisher's exact test [Fis22] gives us the *exact* p-value determining whether the relationship between the variables of the model is significant. This is in contrast to the χ^2-test, which is an *approximation* of the significance. Fisher's exact test is used with small sample groups, while χ^2-test is more suitable for large ones. The formula for Fisher's exact test is

$$p = \frac{\prod (RowTotal_i! \cdot ColumnTotal_j!)}{\prod observed_{ij}! \cdot N!}. \tag{8}$$

Fisher's exact test is similar to the χ^2-test in terms of verification. This is because all that needs to be verified is the frequencies in the contingency table. The same preprocessing on the original data, as the one performed for the χ^2-test is required. The verifiers begin with the verification of the frequencies in the contingency table, and the computation of the total and the marginal totals (same as in χ^2-test). Then, they evaluate equation (8) and check the consistency of the result with the one to be verified. Our detailed algorithm for Fisher's exact test verification is given in Algorithm 9 (Appendix A).

4.9 Privacy-Preserving McNemar's test Verification

McNemar's test [McN47], similarly to Pearson's χ^2-test, gives us an approximation of the significance. This test, given by the following formula

$$\chi^2 = \frac{(observed_{1,2} - observed_{2,1})^2}{observed_{1,2} + observed_{2,1}}, \tag{9}$$

can be only applied to data recorded on a 2×2 contingency table. For McNemar's test, what we verify in the secret shared domain is the frequencies of the contingency table, as in Fisher's exact test and χ^2-test. Hence, the verifiers proceed as described in χ^2-test's verification. Then, they evaluate equation (9) and compare the result with the χ^2-value received for verification. Details are given in Algorithm 10 (Appendix A).

5 Security and Performance Analysis

Our setting lies in the semi-honest model, meaning that the verifiers are assumed to honestly follow the instructions mandated by the protocol, but they wish to learn as much information as possible about the private inputs of the *dealer* (i.e., the hospital). The verifiers are allowed to know all the public inputs given as arguments in the protocols (e.g., the sample group size N of the statistical operation to be verified) and all the public results that they compute. The verifiers are not allowed to learn anything more than the aforementioned, in addition to what can be inferred by the results. Hence, we need to protect the private inputs of the *dealer* and we do so by means of Shamir's Secret Sharing. This way we achieve information-theoretic security, as long as at least $\tau > \frac{n}{2}$ verifiers are honest and do not collude. We assume that there exists pairwise secure channels between the verifiers.

Our performance analysis is based on a proof of concept implementation that we designed to demonstrate the efficiency of our solution. We used real patient data for our experiments to show the applicability of our proposal in practical cases. The aforementioned data concerns patient compliance in a tele-treatment application, where the patients were carrying a monitoring system, measuring their activity and sending them back activity advice, in the form of feedback messages. This dataset consists of 2370 feedback messages of 85 patients that have been analyzed. For more information about the data and the tele-treatment application we refer the reader to [AodJH10].

5.1 Security Analysis

The security requirement that we wish to satisfy is to preserve the confidentiality of the private inputs of the *dealer*, while allowing a certain functionality of the verifiers, enabling the verification of the result of a predefined function. We deal with passive, static adversaries corrupting any minority of the verifiers. Security is modeled using the real vs. ideal

paradigm [Gol04, Section 7.2]. In the real world, the protocol participants execute the protocol interactively and there is an adversary $\mathcal{A}$, having access to all the private inputs and all the messages exchanged among the corrupted parties, as well as the public inputs of both the corrupted and the honest parties. In the ideal world, a protocol is assumed to be executed in the presence of a trusted party, which the protocol participants query and get the appropriate results, based on their predefined functionality. To prove security in the real vs. ideal framework, we show that all adversarial behavior in the real world (where there is no trusted party) can be simulated in the ideal world. In the following, we sketch the construction of such a *simulator* $\mathcal{S}$, while we treat the security of each building block (i.e., subroutine) of our construction separately and the overall security follows by the Composition Theorem for the semi-honest model [Gol04, Theorem 7.3.3].

Observe that in all our algorithms there are only two building blocks that act in the secret shared domain, the SumPub(.) and the InnerPub(.) algorithms. The rest of the computations are performed in the clear. The SumPub(.) algorithm is executed in one communication round. All the messages exchanged in this round are public inputs of the parties (independent of their private inputs). Thus, the views of the adversary and the simulator are exactly the same (i.e., indistinguishable) meaning that SumPub(.) is secure.

The InnerPub(.) algorithm starts with n invocations of the PRZS(.) function, where n is the number of items in each vector. This function was proposed and proven secure in [CDI05]. Next, (in the ideal world) the trusted party computes a share of the inner product per party, and adds to this, his share of a secret shared 0 value ($[0]$), which he obtained from the invocation of the PRZS(.) function. The adversary (in the real world) proceeds similarly, but computes the resulting share for each honest party on random shares (more precisely on shares of uniformly randomly integers), because it has no access to their private inputs. The addition of the secret shared 0 value at this step, restores the uniformly randomness of the shares. Thus, the "fresh" shares do not depend on the private inputs anymore. The aforementioned shares form the public inputs of the parties, which are accessible by both the adversary and the simulator. The indistinguishability of the adversary's (real-world) view from the simulator's (ideal-world) view follows from the fact that the adversary and simulator have identical views of the public inputs and indistinguishable views of the private inputs. The latter holds, because Shamir's shares are perfectly indistinguishable from random integers. Hence, the InnerPub(.) algorithm is proven secure. Having proven secure the SumPub(.) and the InnerPub(.) algorithms, by the composition theorem, the overall security of our protocols is guaranteed.

5.2 Performance Analysis

Our performance analysis presents the execution times of our verification algorithms. The implementation of these algorithms is based on VIFF [VD]. The experiments for timing our verification algorithms were conducted on an Intel(R) Core(TM) i3-2350M processor, at 2.3 GHz, with 4.00 GB RAM and Windows 7 64-bit operating system. We have conducted all tests on localhost, with 3 verifiers, and the network latency has not been taken into account. In VIFF, the prime p, determining the size of the field $\mathbb{Z}_p$, is selected to be

greater than $(2^{l+1} + 2^{l+k+1})$, where l is the maximum bit length of the inputs, and k is the statistical security parameter. These values are by default set to $l = 32$ and $k = 30$. For reasons of uniformity of the execution time results we use these default values for all our experiments. To handle fixed point numbers occurring in our setting, we scale them up to a precision of five decimal digits and then treat them as integers. The selection of $l = 32$ bits is large enough to handle this scaling both for our inputs and for our outputs.

In addition to the computational cost, determined by the number of variables, the number of entries per variable and the size p (of ~ 64 bits) of the field $\mathbb{Z}_p$, the communication cost also plays an important role in the overall performance. Recall that in our algorithms, the communication cost is brought only by the invocations of SumPub(.) and InnerPub(.), which are the only building blocks acting in the secret shared domain. The communication cost of each such invocation remains constant in all our algorithms, as for both subroutines only one aggregated value is sent, the size of which is upper bounded by the size of the field $\mathbb{Z}_p$, which is also constant and equal to 8 bytes (corresponding to the 64 bits of the prime p) per invocation, per verifier. Hence, our communication cost solely depends on the number of interactive rounds (i.e., the number of SumPub(.) and InnerPub(.) invocations). In the following, we deal with the round complexity of each statistic separately.

Mean and Variance: For the verification of the mean and variance we calculated those two statistics on the ages of 84 patients. One patient (out of the originally 85 patients) was excluded from the specific tests, because his age value was missing. The variable that influences the runtime is the number of patients $N = 84$ and the number of communication rounds is 1 and 2, for the mean and variance, respectively. The performance results of the Mean and Variance verification algorithms are presented in Table 1. These two tests do not scale perfectly linearly, due to their very small execution times.

	Mean	**Variance**
84 patients	43.5 ms	43.7 ms
168 patients	45.1 ms	49.7 ms
252 patients	45.8 ms	49.9 ms

Table 1: Performance of Mean and Variance Verification

ANOVA, Simple Linear Regression, Student's and Welch's t-tests: We conducted Welch's t-test on the time elapsed between receiving a message and reading it, versus patient compliance to that message. We have split our dataset of 2370 messages based on whether the patient complied to the message ($N_1 = 1404$) or not ($N_2 = 966$). Welch's t-test in this case, identifies whether there is a significant relationship between the compliance to a feedback message and the time elapsed between receiving it and reading it. It completes its execution in 4 rounds. For ANOVA we used the same time variable, versus the diagnosis (categorical variable taking 4 distinct diagnosis values) for the patient reading the feedback message in question. The dataset of 2370 messages is split based on the 4 different diagnoses to $N_1 = 953, N_2 = 524, N_3 = 365, N_4 = 528$ and requires 8 communication rounds. For simple linear regression, we examine the significance of the relationship between the aforementioned time variable (dependent variable), and the age of each patient (independent variable). We have excluded from our dataset the messages

concerning a patient for whom the age value was missing, and the size of our sample is $N = 2276$, while the number of required communication rounds is 4.

Our dataset does not contain two equal sized groups on which we can perform Student's t-test. Thus, we have excluded these runtimes from our performance results. Given the similarity of these two tests, their corresponding runtimes for the verification algorithms are also expected to be similar. Our performance results for Welch's t-test and F-test are summarized in Table 2; for regression they are given in Table 3. These tests scale linearly in the number of inputs (see Tables 2 and 3), as we have shown by doubling and tripling our dataset, and executing the algorithms on the augmented datasets.

	Welch's t-test	**F-test**
2370 msgs	165.5 ms	171.6 ms
4740 msgs	291.9 ms	315.1 ms
7110 msgs	404.1 ms	479.0 ms

Table 2: Performance of Welch's t-test and F-test Verification

	Regression
2276 msgs	304.3 ms
4552 msgs	586.4 ms
6828 msgs	884.6 ms

Table 3: Performance of Simple Linear Regression Verification

Chi-Squared test, Fisher's exact test and McNemar's test: We performed χ^2-test on the compliance to each of the 2370 feedback messages versus the diagnosis, to determine whether the data follows the χ^2 distribution. For Fisher's exact test, we split our dataset based on the gender of the 85 patients versus the diagnosis. We also performed and verified McNemar's test on the compliance to each of the 2370 feedback messages versus the feedback type (i.e., encouraging or discouraging message). Both the χ^2-test and Fisher's exact test verification require 8 communication rounds each, while McNemar's test requires 4 rounds. The performance results for χ^2-test and McNemar's test verification algorithms are given in Table 4, while the corresponding results for Fisher's test are presented in Table 5. As expected from our experimental setup, the execution time of the verification algorithms scales linearly in the number of input data.

	Chi-Squared	**McNemar's**
2370 msgs	207.3 ms	138.8 ms
4740 msgs	397.7 ms	238.8 ms
7110 msgs	594.4 ms	352.7 ms

Table 4: Performance of Chi-Squared and McNemar's test Verification

	Fisher's test
85 patients	53.3 ms
170 patients	57.7 ms
255 patients	70.3 ms

Table 5: Performance of Fisher's exact test Verification

6 Conclusion

We deal with privacy-preserving verification of statistics in clinical research, where the prover does not wish to disclose information about the inputs to the verifier. This is reciprocal to scenarios that previous works address, where the verifier does not wish to disclose

information about the inputs to the prover. We demonstrate that in the clinical research scenario under consideration, privacy-preserving verification of statistics can be performed so efficiently, that it can be applied in practice.

Acknowledgements: This work has been done in the context of the THeCS project which is supported by the Dutch national program COMMIT. We would like to thank Lorena Montoya and Job van der Palen for their help in statistics and clinical research.

References

[ABC+12] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters. Computing on Authenticated Data. In *TCC*, pages 1–20. Springer, 2012.

[AIK10] B. Applebaum, Y. Ishai, and E. Kushilevitz. From Secrecy to Soundness: Efficient Verification via Secure Computation. In *ICALP*, pages 152–163. Springer, 2010.

[AodJH10] H. Akker op den, V. Jones, and H. Hermens. Predicting Feedback Compliance in a Teletreatment Application. In *ISABEL*, pages 1–5. IEEE, 2010.

[BCD+09] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. I. Schwartzbach, and T. Toft. Secure Multiparty Computation Goes Live. In *FC*, pages 325–343. Springer, 2009.

[BF11a] D. Boneh and D. Freeman. Homomorphic Signatures for Polynomial Functions. In *EUROCRYPT*, pages 149–168. Springer, 2011.

[BF11b] D. Boneh and D. Freeman. Linearly Homomorphic Signatures over Binary Fields and New Tools for Lattice-Based Signatures. In *PKC*, pages 1–16. Springer, 2011.

[BGV11] S. Benabbas, R. Gennaro, and Y. Vahlis. Verifiable Delegation of Computation over Large Datasets. In *CRYPTO*, pages 111–131. Springer, 2011.

[BLW08] D. Bogdanov, S. Laur, and J. Willemson. Sharemind: A Framework for Fast Privacy-Preserving Computations. In *ESORICS*, pages 192–206. Springer, 2008.

[BSMD10] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos. SEPIA: Privacy-Preserving Aggregation of Multi-Domain Network Events and Statistics. In *USENIX*, pages 223–240. USENIX Assoc., 2010.

[CDI05] R. Cramer, I. Damgård, and Y. Ishai. Share Conversion, Pseudorandom Secret-Sharing and Applications to Secure Computation. In *TCC*, pages 342–362. Springer, 2005.

[CHd10] O. Catrina and S. Hoogh de. Secure Multiparty Linear Programming Using Fixed-Point Arithmetic. In *ESORICS*, pages 134–150. Springer, 2010.

[DA01] W. Du and M. J. Atallah. Privacy-Preserving Cooperative Statistical Analysis. In *ACSAC*, pages 102–110. IEEE, 2001.

[DF97] G. T. Duncan and S. E. Fienberg. Obtaining Information while Preserving Privacy: A Markov Perturbation Method for Tabular Data. In *JSM*, pages 351–362. IOS Press, 1997.

[DHC04] W. Du, Y. S. Han, and S. Chen. Privacy-Preserving Multivariate Statistical Analysis: Linear Regression and Classification. In *SIAM SDM*. Lake Buena Vista, Florida, 2004.

[DN04] C. Dwork and K. Nissim. Privacy-Preserving Datamining on Vertically Partitioned Databases. In *CRYPTO*, pages 134–138. Springer, 2004.

[Ens] M. Enserink. Stapel Affair Points to Bigger Problems in Social Psychology; http://news.sciencemag.org/scienceinsider/2012/11/final-report-stapel-affair-point.html.

[Fan09] D. Fanelli. How Many Scientists Fabricate and Falsify Research? A Systematic Review and Meta-Analysis of Survey Data. *PLOS ONE*, 4(5):e5738, 2009.

[Fie09] A. Field. *Discovering Statistics Using SPSS*. Sage Publications Limited, 2009.

[Fis22] R. A. Fisher. On the Interpretation of χ^2 from Contingency Tables, and the Calculation of P. *J. R. Stat. Soc.*, pages 87–94, 1922.

[GGP10] R. Gennaro, G. Gentry, and B. Parno. Non-Interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers. In *CRYPTO*, pages 465–482. Springer, 2010.

[Gol04] O. Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, 2004.

[GRR98] R. Gennaro, M. O. Rabin, and T. Rabin. Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography. In *PODC*, pages 101–111. ACM, 1998.

[HR10] M. Hardt and G. N. Rothblum. A Multiplicative Weights Mechanism for Privacy-Preserving Data Analysis. In *FOCS*, pages 61–70. IEEE, 2010.

[JMSW02] R. Johnson, D. Molnar, D. Song, and D. Wagner. Homomorphic Signature Schemes. In *CT-RSA*, pages 204–245. Springer, 2002.

[KLSR09] A. F. Karr, X. Lin, A. P. Sanil, and J. P. Reiter. Privacy-Preserving Analysis of Vertically Partitioned Data Using Secure Matrix Products. *JOS*, 25(1):125, 2009.

[McN47] Q. McNemar. Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages. *Psychometrika*, 12(2):153–157, 1947.

[Md09] J. E. Muth de. Overview of Biostatistics Used in Clinical Research. *AJHP*, 66(1):70–81, 2009.

[Mis] Misconduct in Science: An Array of Errors; http://www.economist.com/node/21528593.

[OS08] B. R. Overholser and K. M. Sowinski. Biostatistics Primer: Part 2. *NCP*, 23(1):76–84, 2008.

[Pea00] K. Pearson. X. On the Criterion that a Given System of Deviations from the Probable in the Case of a Correlated System of Variables Is Such that It Can Be Reasonably Supposed to Have Arisen from Random Sampling. *Lond. Edinb. Dubl. Phil. Mag.*, 50(302):157–175, 1900.

[PRV12] B. Parno, M. Raykova, and V. Vaikuntanathan. How to Delegate and Verify in Public: Verifiable Computation from Attribute-Based Encryption. In *TCC*, pages 422–439. Springer, 2012.

[PST13] C. Papamanthou, E. Shi, and R. Tamassia. Signatures of Correct Computation. In *TCC*, pages 222–242. Springer, 2013.

[RBG+00] J. Ranstam, M. Buyse, S. L. George, S. Evans, N. L. Geller, B. Scherrer, E. Lesaffre, G. Murray, L. Edler, J. L. Hutton, T. Colton, and P. Lachenbruch. Fraud in Medical Research: An International Survey of Biostatisticians. *Control clin trials*, 21(5):415–427, 2000.

[SCD+08] R. Sparks, C. Carter, J. B. Donnelly, C. M. O'Keefe, J. Duncan, T. Keighley, and D. McAullay. Remote Access Methods for Exploratory Data Analysis and Statistical Modelling: Privacy-Preserving Analytics. *Comput Meth Prog Bio*, 91(3):208–222, 2008.

[Sha79] A. Shamir. How to Share a Secret. *Comm. of the ACM*, 22(11):59–98, 1979.

[Swe02] L. Sweeney. k-Anonymity: A Model for Protecting Privacy. *Uncertain Fuzz*, 10(5):557–570, 2002.

[THH+09] B. Thompson, S. Haber, W. G. Horne, T. Sander, and D. Yao. Privacy-Preserving Computation and Verification of Aggregate Queries on Outsourced Databases. In *PETS*, pages 185–201. Springer, 2009.

[Tho] THORAX - An International Journal Of Respiratory Medicine; http://thorax.bmj.com/.

[VD] Team VIFF Developement. The Virtual Ideal Functionality Framework; http://viff.dk/.

[ZBT07] K. Zellner, C. J. Boerst, and W. Tabb. Statistics Used in Current Nursing Research. *Nurs Educ*, 46(2):55–59, 2007.

A Verification Algorithms

Algorithm 1 $v \leftarrow \mathsf{VrfMean}([\mathbf{x}], N, \bar{x})$

Input: [**x**] = secret shared vector **x**, N = size of sample group, $\bar{x}$ = calculated mean value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
for all P_j **do**
  if $(N \cdot \bar{x}) \neq \mathsf{SumPub}([\mathbf{x}])$ **then**
    return 0
return 1

Algorithm 2 $v \leftarrow \mathsf{VrfVariance}([\mathbf{x}], \bar{x}, N, S^2)$

Input: [**x**] = secret shared vector **x**, $\bar{x}$ = mean value of sample group x, N = size of sample group, S^2 = calculated variance to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if $\mathsf{VrfMean}([\mathbf{x}], N, \bar{x}) == 0$ **then**
    return 0
for all P_j **do**
    if $((N-1) \cdot S^2) \neq \mathsf{InnerPub}(([\mathbf{x}] - \bar{x}), ([\mathbf{x}] - \bar{x}))$ **then**
        return 0
return 1

Algorithm 3 $v \leftarrow \mathsf{VrfS\text{-}T\text{-}test}([\mathbf{x}], [\mathbf{y}], \bar{x}, \bar{y}, N, S_x^2, S_y^2, t)$

Input: [**x**] = secret shared vector **x**, [**y**] = secret shared vector **y**, $\bar{x}$ = mean value of sample group x, $\bar{y}$ = mean value of sample group y, N = size of sample group, S_x^2 = variance of sample group x, S_y^2 = variance of sample group y, $t = t$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if $\mathsf{VrfVariance}([\mathbf{x}], \bar{x}, N, S_x^2) == 0$ **or** $\mathsf{VrfVariance}([\mathbf{y}], \bar{y}, N, S_y^2) == 0$ **then**
    return 0
for all P_j **do**
    if $(\bar{x} - \bar{y}) / (\sqrt{\frac{S_x^2 + S_y^2}{N}}) \neq t$ **then**
        return 0
return 1

Algorithm 4 $v \leftarrow \mathsf{VrfW\text{-}T\text{-}test}([\mathbf{x}], [\mathbf{y}], \bar{x}, \bar{y}, N_x, N_y, S_x^2, S_y^2, t)$

Input: [**x**] = secret shared vector **x**, [**y**] = secret shared vector **y**, $\bar{x}$ = mean value of sample group x, $\bar{y}$ = mean value of sample group y, N_x = size of sample group x, N_y = size of sample group y, S_x^2 = variance of sample group x, S_y^2 = variance of sample group y, $t = t$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if $\mathsf{VrfVariance}([\mathbf{x}], \bar{x}, N_x, S_x^2) == 0$ **or** $\mathsf{VrfVariance}([\mathbf{y}], \bar{y}, N_y, S_y^2) == 0$ **then**
    return 0
for all P_j **do**
    if $(\bar{x} - \bar{y}) / (\sqrt{\frac{S_x^2}{N_x} + \frac{S_y^2}{N_y}}) \neq t$ **then**
        return 0
return 1

Algorithm 5 $v \leftarrow$ VrfF-test$(K, [\mathbf{x}][K], \bar{x}[K], N[K], S^2[K], F)$

Input: K = number of groups, [**x**] = table of K secret shared vectors $\mathbf{x}[K]$, $\bar{x}[K]$ = mean values of K sample groups x, $N[K] = K$ sizes of the sample groups, $S^2[K]$ = variances of the K sample groups, $F = F$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
for $k = 1, ..., K$ **do**
  if VrfVariance$([\mathbf{x}]_k, \bar{x}_k, N_k, S_k^2) == 0$ **then**
    return 0
for all P_j **do**
  if $(\sum_{k=1}^{K} \frac{N_k(\bar{x}_k - \frac{\sum_{k=1}^{K} \bar{x}_k}{K})^2}{K-1}) / (\sum_{k=1}^{K} \frac{([\mathbf{x}]_k - \bar{x}_k)^2}{\sum_{k=1}^{K} N_k - K}) \neq F$ **then**
    return 0
return 1

Algorithm 6 $v \leftarrow$ VrfSimpleLinearRegression$([\mathbf{x}], [\mathbf{y}], \bar{x}, \bar{y}, \alpha, \beta, N)$

Input: [**x**] = secret shared vector **x**, [**y**] = secret shared vector **y**, $\bar{x}$ = mean value of sample group x, $\bar{y}$ = mean value of sample group y, $\alpha = \alpha$-value to be verified, $\beta = \beta$-value to be verified, N = size of sample group
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if VrfMean$([\mathbf{x}], N, \bar{x}) == 0$ **or** VrfMean$([\mathbf{y}], N, \bar{y}) == 0$ **then**
  return 0
for all P_j **do**
  if $\frac{\text{InnerPub}(([\mathbf{x}]-\bar{x}),([\mathbf{y}]-\bar{y}))}{\text{InnerPub}(([\mathbf{x}]-\bar{x}),([\mathbf{x}]-\bar{x}))} \neq \beta$ **or** $(\bar{y} - \beta \cdot \bar{x}) \neq \alpha$ **then**
    return 0
return 1

Algorithm 7 $v \leftarrow$ VrfContingency$([\mathbf{x}[C1]], [\mathbf{y}[C2]], C1, C2, F[C1, C2])$

Input: [**x**] = $C1 \cdot N$ secret shared bits of the unary representation of vector **x**, [**y**] = $C1 \cdot N$ secret shared bits of the unary representation of vector **y**, $C1$ = number of possible values under "cause" of the experiment, $C2$ = number of possible values under "effect" of the experiment, $F[C1, C2]$ = table of size $C1 \cdot C2$ containing the frequencies observed (i.e., inner part of the contingency table)
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
for $k = 1, ..., C1$ **do**
  for $l = 1, ..., C2$ **do**
    $VF[k, l] \leftarrow$ InnerPub$([\mathbf{x}_k], [\mathbf{y}_l])$
    for all P_j **do**
      if $VF[k, l] \neq F[k, l]$ **then**
        return 0
return 1

Algorithm 8 $v \leftarrow$ VrfChi-squaredTest$([\mathbf{x}[C1]], [\mathbf{y}[C2]], C1, C2, F[C1, C2], \chi^2)$

Input: [**x**] (resp. [**y**]) = bitwise secret shared vector **x** (resp. **y**), where each entry has $C1$ (resp. $C2$) bits, $C1$ = number of possible values under "cause" of the experiment, $C2$ = number of possible values under "effect" of the experiment, $F[C1, C2]$ = table of size $C1 \cdot C2$ containing the frequencies observed, $\chi^2 = \chi^2$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if VrfContingency$([\mathbf{x}[C1]], [\mathbf{y}[C2]], C1, C2, F[C1, C2]) == 0$ **then**
  return 0
for all P_j **do**
  $model_{k,l} \leftarrow (\sum_{l=1,k=1}^{C2,C1} F_{k,l} \cdot \sum_{k=1,l=1}^{C1,C2} F_{k,l})/(\sum_{k=1}^{C1} RowTotal_k)$
  if $\sum_{k=1,l=1}^{C1,C2} \frac{(F_{k,l} - model_{k,l})^2}{modelk,l} \neq \chi^2$ **then**
    return 0
return 1

Algorithm 9 $v \leftarrow$ VrfFisher'sTest$([\mathbf{x}[C1]], [\mathbf{y}[C2]], C1, C2, F[C1, C2], p)$

Input: [**x**] (resp. [**y**]) = bitwise secret shared vector **x** (resp. **y**), where each entry has $C1$ (resp. $C2$) bits, $C1$ = number of possible values under "cause" of the experiment, $C2$ = number of possible values under "effect" of the experiment, $F[C1, C2]$ = table of size $C1 \cdot C2$ containing the frequencies observed, $p = p$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if VrfContingency$([\mathbf{x}[C1]], [\mathbf{y}[C2]], C1, C2, F[C1, C2]) == 0$ **then**
  return 0
for all P_j **do**
  $VpUpper \leftarrow \prod_{k=1}^{C1} \sum_{l=1,k=1}^{C2,C1} F_{k,l}! \cdot \prod_{l=1}^{C2} \sum_{k=1,l=1}^{C1,C2} F_{k,l}!$ #numerator of the p fraction
  $VpLower \leftarrow \prod_{k=1,l=1}^{C1,C2} F_{k,l}! \cdot \sum_{k=1}^{C1} \sum_{l=1,k=1}^{C2,C1} F_{k,l}!$ #denominator of the p fraction
  if $\frac{VpUpper}{VpLower} \neq p$ **then**
    return 0
return 1

Algorithm 10 $v \leftarrow$ VrfMcNemar'sTest$([\mathbf{x}[2]], [\mathbf{y}[2]], F[2, 2], \chi^2)$

Input: [**x**] (resp. [**y**]) = bitwise secret shared vector **x** (resp. **y**), where each entry has 2 bits, $F[2, 2]$ = 2×2 table of frequencies, $\chi^2 = \chi^2$-value to be verified
Output: v = 0 or 1; 0 $\rightarrow$ unsuccessful verification, 1 $\rightarrow$ successful verification
if VrfContingency$([\mathbf{x}[2]], [\mathbf{y}[2]], 2, 2, F[2, 2]) == 0$ **then**
  return 0
for all P_j **do**
  if $\frac{(F_{1,2} - F_{2,1})^2}{F_{1,2} + F_{2,1}} \neq \chi^2$ **then**
    return 0
return 1