Network Information Hiding

Terminology, Taxonomy, Methodology and Countermeasures

Steffen Wendzel

Habilitation Thesis

FERNUNIVERSITÄT IN HAGEN

FAKULTÄT FÜR MATHEMATIK & INFORMATIK

HABILITATIONSSCHRIFT

Network Information Hiding: Terminology, Taxonomy, Methodology and Countermeasures

Autor:
Dr. rer. nat. STEFFEN WENDZEL

November 2019

"In my own field, for example, it once was possible for a grad student to learn just about everything there was to know about computer science. [...] Nowadays the subject is so enormous, nobody can hope to cover more than a tiny portion of it."

Donald E. Knuth

FERNUNIVERSITÄT IN HAGEN

Fakultät für Mathematik und Informatik

Abstract

Habilitation Thesis

Network Information Hiding: Terminology, Taxonomy, Methodology and Countermeasures

by Dr. rer. nat. STEFFEN WENDZEL

Network information hiding is the research discipline that deals with the concealment of network transmissions or their characteristics. It serves as an umbrella for multiple research domains, namely network covert channel research, network steganography research, and traffic obfuscation research. The focus of this thesis lies primarily on network steganography and network covert channel research.

This thesis was motivated by the fact that network information hiding requires a better scientific foundation. When the author started to work on this thesis, scientific re-inventions of hiding techniques were common (similar or equal techniques were published under different names by different scientific sub-communities). This is, at least partially, rooted in the non-unified terminology of the domain, and, linked to the quotation of Donald Knuth on the previous page, in the sheer fact that the ever increasing number of publications in the domain is hardly knowable. Moreover, experimental results and descriptions for hiding techniques are hardly comparable as there is no unified standard for describing them. This is a contrast to other scientific domains, such as Chemistry, were (de facto) standards for experimental descriptions are common. Another problem is that experimental results are not replicated while other scientific domains have shown that replication studies are a necessity to ensure the quality of scientific results. Finally, there is an imbalance between known hiding techniques and their countermeasures: not enough countermeasures are known to combat all known hiding techniques.

To address these issues, this thesis motivates and proposes methodological adjustments in network information hiding and lays the foundation for an improved fundamental terminology and taxonomy.

Moreover, hiding techniques are surveyed and summarized in the form of abstract descriptions, called *hiding patterns*, which form an extensible taxonomy. These hiding patterns are then used as a tool to evaluate the novelty of research contributions in a scientific peer-review process. Afterwards, this thesis addresses the problem of inconsistent descriptions of hiding techniques by proposing a unified description method for the same, including hiding patterns as a core component of every description. This thesis also introduces the *WoDiCoF* framework to perform replication studies.

Afterwards, the concept of *countermeasure variation* is introduced to address the problem of not having countermeasures available for certain hiding patterns. Finally, the proposed pattern-based taxonomy is enhanced to demonstrate the extensibility of the taxonomy and to integrate payload-based hiding techniques which were not foreseen in the earlier version of the taxonomy.

Acknowledgements

First, I like to thank Prof. Dr. Jörg Keller from the Faculty of Mathematics and Computer Science for his guidance during the last ten years, which first led to the completion of my PhD in 2013 and afterwards to the support of my habilitation thesis, not to mention several joint efforts, such as the organization of workshops and journal special issues.

I started the research that later led to this thesis back in 2014 as a PostDoc at the Department of Cyber Security, Fraunhofer FKIE, Bonn. I like to thank Prof. Dr. Michael Meier for giving me the freedom to conduct the research that I desired to work on during this time in his department and also to involve me in several additional academic opportunities, such as conference organization and creation of funding proposals.

I also like to thank all my co-authors for the highly disciplined and efficient joint-work that we conducted during the last years, especially Luca Caviglione, Wojciech Mazurczyk and Sebastian Zander.

My thankfulness is also well-deserved by my family members who supported me along this way. Especially, I like to thank my fiancé Carolin. She understood the importance that research has for me from day one of our relationship and encouraged me to pursue it.

Contents

List of Figures

List of Tables

List of Abbreviations

AH	Authentication Header
ARP	Address Resolution Protocol
AUC	Area Under the ROC Curve
BACnet	Building Automation Control Networks
CC	Covert Channel
CCEAP	Covert Channel Educational Analysis Protocol
CoAP	Constrained Application Protocol
CR	Covert Receiver
CS	Covert Sender
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CSPRNG	Cryptographically Secure PRNG
CUING	Criminal Use of Information Hiding (Initiative)
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DoS	Denial of Service
DSSS	Direct Sequence Spread Spectrum
ESP	Encapsulating Security Payload
FN	False Negative
FP	False Positive
FPR	False-positive Rate
FTP	File Transfer Protocol
HDFS	Hadoop Distributed Filesystem
HTTP	Hyper-text Transfer Protocol
ICMP	Internet Control Message Protocol
IM	Intermediate Node
IAT	Inter-arrival Time
IDS	Intrusion Detection System
IPG	Inter-packet Gap
IP(v4)	Internet Protocol, version 4
IPv6	Internet Protocol, version 6
ISN	Initial Sequence Number
LAN	Local Area Network
LEA	Law-enforcement Agency
LSB	Least Significant Bit
MitM	Man-in-the-Middle
MLS	Multilevel Security
MQTT	Message Queuing Telemetry Transport
NNTP	Network News Transfer Protocol
NZIX	New Zealand Internet Exchange
OSI	Open Systems Interconnected
PCAP	Packet Capture (file format)
PDU	Protocol Data Unit
PLML	Pattern Language Markup Language

PRNG	Pseudo-random Number Generator
RFC	Request for Comments
SCC	Storage Covert Channel
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
TCP	Transmission Control Protocol
TCC	Timing Covert Channel
TN	True Negative
TP	True Positive
TTL	Time to Live
UDP	User Datagram Protocol
VPN	Virtual Private Network
WoDiCoF	Worms Distributed Covert Channel Detection Framework
XMPP	Extensible Messaging and Presence Protocol

Dedicated to my former advisors and teachers

Part I

Introduction

Chapter 1

Introduction

Network information hiding is the discipline that deals with the covert transfer of data through a computer network and with the detection, limitation and prevention of such transfers. First techniques for stealthy network transmissions arose in the 1980's. Today, hundreds of hiding techniques exist and represent manifold ways to signal secret data over all relevant network protocols, be it IPv4, IPv6, TCP, BACnet or CoAP.

Research work on network information hiding is usually specifically tailored to these network protocols. For instance, a typical research paper could show that covert channels in CoAP exist. In this protocol-specific focus, so-called "re-inventions" take place, i.e. the same forms of data hiding techniques are applied to different protocols in a recurring manner while being falsely presented as "new". Moreover, does the non-unified terminology in this young domain support re-inventions under different terms, i.e. the same ideas are eventually published multiple times using different wording such as protocol tunneling, network steganography, network covert channel, data hiding etc. These varying terms are then applied to different protocols. For instance, one paper could show how data is "tunneled" through the least significant bits of the IPv4 *TTL* field while another paper might analyze a *covert channel* exploiting the least significant bits of the IPv6 *Hop Count* field, which is essentially the same approach.

In an interview during the Swiss TV show *Sternstunde Philosophie*, Nassim Taleb once mentioned that a laptop might not be a device that we use in thirty years from that point in time, but a typical glass of water will still be used in thirty years and was already used centuries ago. This is because the glass represents a long-lasting solution to a re-occurring problem. In software engineering such an idea is called a *design pattern*. The desire behind this thesis is provide results that are more like the glass of water: our central goal is to limit (optimally prevent) re-inventions in the network information hiding domain on the basis of *hiding patterns*. We call hiding patterns the fundamental concepts that describe how data can be hidden in transmissions. The intent of specifying and using hiding patterns is that they describe hiding approaches generic enough to help withstanding the designs of coming generations of communication protocols and thus help keeping the terminology in the domain consistent, i.e. limiting the mentioned re-inventions. Moreover, does this thesis propose other approaches, such as a unified description of hiding techniques to render experimental results comparable, and a framework to assess the novelty of research

contributions based on hiding patterns during academic peer reviews. In addition, countermeasures with a focus on patterns are also introduced.

On a more abstract level, the goal of this thesis can be summarized as *fostering improvements of the scientific methodology within the domain of network information hiding by introducing and using hiding patterns.* The following section summarizes all key contributions.

1.1 Contributions

The following points summarize the major contributions which are each represented by a separate chapter.

- **Improvement of the Network Information Hiding Terminology**
  After introducing the research domain, fundamental enhancement on terminology and taxonomy are presented, especially to address terminological inconsistencies.
- **Motivating Improvement of the Methodology**
  Methodological improvements are discussed and motivated under the umbrella of the Science 2.0 paradigm, such as experimental replication studies. These aspects are then addressed in the following chapters.
- **Introduction of Hiding Patterns**
  A new taxonomy for hiding methods on the basis of hiding patterns is introduced. A hiding pattern represents the core idea of a family of hiding techniques that are based on a common principle (e.g., the modulation of packet sizes). To achieve this, 109 hiding methods were surveyed and categorized. Additional pattern-based concepts, such as *pattern hopping* and *pattern variation* are introduced.
- **Pattern-based Novelty Evaluation**
  Publications often introduce already known or slightly varied versions of existing hiding techniques, often under different names. For reviewers, the novelty of such methods is often not clearly accessible, leading to the acceptance of ideas that are *considered* to be novel. With hiding patterns, scientists are able to show how new hiding techniques are fundamentally different from existing ones (by explaining how a new hiding method cannot be represented by one of the patterns) or that they indeed belong to an already existing idea. We present a pattern-based framework to address the evaluation of novelty during academic peer review. The framework helps preventing scientific re-inventions and terminological inconsistencies.
- **Unified Description Method for Hiding Techniques**
  Scientific publications present hiding techniques in unique ways, rendering results difficult or even impossible to compare. A unified description method for new hiding methods is introduced which makes future's hiding methods comparable if the unified description method is applied by the scientific community.

- **Replication Framework and Exemplary Replication Study**
  We present a framework that was used to conduct the first experimental replication of a highly cited covert channel detection algorithm that was presented in an ACM TISSEC article with 130+ citations (an extended version of an ACM CCS paper with 470+ citations). Our experiment shows that, although they are not performed in network information hiding, replication studies can improve the understanding of methods and should become a core element of the scientific methodology within the research domain.
- **Introduction of Countermeasure Variation**
  Countermeasures are designed as tailored methods to combat specific covert channels. A drawback of these methods is that when new hiding techniques are introduced or existing ones are modified, there are no countermeasures available to combat these new or modified hiding techniques. We introduce countermeasure variation as a tool to modify existing countermeasures tailored for one hiding pattern to work with *another* hiding pattern. We exemplify the feasibility of countermeasure variation for two countermeasures (compressibility score and ϵ-similarity) and different hiding patterns.
- **Improvement of the Hiding Pattern Taxonomy**
  Finally, we provide improvements of the hiding patterns taxonomy, especially to extend the number of known hiding patterns and to provide a clear taxonomy for distributed hiding methods. This was also done to prove that hiding patterns are robust enough to manage extensions and adjustments by the community in order to reflect scientific advancements.

Several of the original contributions of this thesis were conducted jointly with other authors, including computer science students in the context of their theses. For this reason, each chapter contains a list of papers and their authors that served as the basis for the particular chapter. However, the core publications for this thesis feature this thesis' author as first or – at the very least – second author. The appendix of this thesis provides an overview of all publications of the author and allows the reader to assess which papers were first and second author papers. Each papers' content was edited to fit into the structure of this thesis. Moreover, several improvements, extensions and updates were applied to the original publications' contents, resulting in additional upcoming publications that are currently in preparation, under review, or in press.

1.2 Thesis Overview

This thesis is designed to provide the reader with a comprehensive coverage of hiding pattern-based terminology, taxonomy, and methodology. The structure of thesis is visualized in Figure 1.1.

Chapters 2 to 4 (**light gray**) provide the foundation for this thesis. Chapter 2 introduces the scientific domain of network information hiding (especially network steganography) and contributes to the fundamental terminology

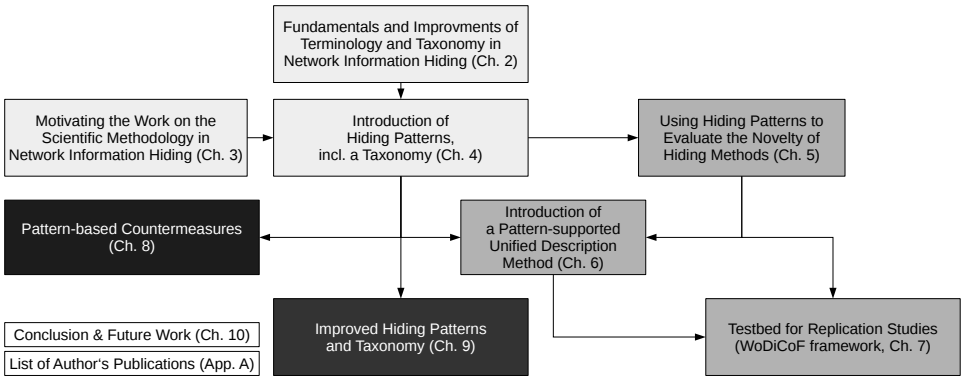

FIGURE 1.1: Structure of this thesis and relation of chapters

and taxonomy for network information hiding; it is the result of excessive discussions and agreements within the community. Chapter 3 motivates the advancement of a scientific methodology which is the primary subject of this thesis, i.e., explaining the need for experimental replications and a consistent terminology. The core tool (*hiding patterns* and their taxonomy) of this thesis are provided by Chapter 4. Throughout the whole document, patterns are applied as a tool to improve the scientific methodology in network information hiding.

Chapters 5 to 7 (**gray**) present a framework for a new scientific methodology in network information hiding: Chapter 5 presents an approach that describes how the novelty of covert channel papers that present new hiding techniques can be evaluated during academic peer review in a way that scientific re-inventions and terminological inconsistencies can be addressed. To finally make hiding techniques comparable, a unified description method for them is introduced in Chapter 6. Chapter 7 presents a framework for the replication of network covert channel experiments and provides the final puzzle piece for the scientific methodology introduced in this thesis.

Chapters 8 and 9 (**dark gray**) provide extended ideas based upon the previous chapters: Chapter 8 proposes the idea of pattern-based countermeasures and analyzes its feasibility with experimental studies. The existing hiding patterns and related taxonomy are updated and extended by additional patterns and aspects of distributed hiding methods in Chapter 9.

Finally, Chapter 10 summarizes this thesis and discusses potential future work. Appendix A contains a list of the author's academic publications (and related awards).

Part II

Fundamental Terminology, Methodology & Patterns

Chapter 2

Improving Fundamental Terminology and Taxonomy for Network Information Hiding

Abstract This chapter starts with a brief introduction to network information hiding. Developments in the related sub-disciplines are overlapping, fostering scientific re-inventions due to a inconsistent terminology. For this reason, this chapter presents a unified terminology of *basic terms* in network information hiding. In comparison to other chapters of this thesis, this chapter is based on intensive joint-work with a heterogeneous community as the proposal of an improved fundamental terminology requires its acceptance by other scientists as well as their involvement.

Originally published: Wendzel, Mazurczyk, Caviglione, and Meier (2014b); Mazurczyk, Wendzel, Zander, Houmansadr, and Szczypiorski (2016a, Chapter 2*); Mazurczyk, Wendzel, Villares, and Szczypiorski (2016b); Mazurczyk and Wendzel (2018); Cabaj, Caviglione, Mazurczyk, Wendzel, Woodward, and Zander (2018a)

* Content of the publication was extended and improved for this thesis. This chapter is – to its largest extend – based on Chapter 2 of (Mazurczyk et al., 2016a); additional results of follow-up works and improvements were merged into this chapter. However, several aspects, e.g. the terminological unification of "deep hiding techniques" were left out as they are not required for the following chapters.

2.1 Introduction

To understand the concept of *network information hiding* let us consider the following real-life scenario: a crowded airport. Such an airport is like a router in a communication network and passengers are like packets – every day a lot of passengers are passing through an airport and they are traveling in many different directions. Also like packets in routers the passengers are "inspected" before entering a plane, e.g., by using increasingly popular full-body scanners which are able to detect illegal objects on the passenger's body without making actual physical contact or without removing clothes.

However, even though this technology is continuously improved, still from time to time the security is breached and some illegal objects are smuggled into the plane. This is achieved by placing contraband in locations on the body where it is invisible on the body-scanner monitor.

To be able to deceive such a scanner the thorough knowledge of the device's inner functioning and limitations is required. This allows to identify potential vulnerabilities that can be exploited for illicit purposes.

In general, the same principle is utilized in network information hiding. To successfully transmit data in a covert manner, a solution to embed secret data into a network traffic flow must be found where the resulting modifications are not "visible" to network devices as well as end users.

Based on the aim to be achieved, information hiding techniques can be used by criminals/terrorists and other malicious actors for the following purposes:

- **As a mean for covert storage:** To hide secret data in such a way that no one besides the owner is authorized to discover its location and retrieve it. In other words, the aim is to not reveal existence of the stored secret to any undesired party. This way criminals/terrorists can store their secret data in a hidden manner.
- **As a covert communication tool:** To communicate messages with the aim of keeping some aspect of their exchange secret. Criminals/terrorists can use information hiding to covertly exchange their confidential data.
- **As a data exfiltration technique:** Cybercriminals/insiders can use it to steal/exfiltrate confidential data.
- **As a mean for covert malware communication:** Finally, malware can be equipped with information hiding techniques to become stealthier while residing on the infected host and/or while communicating with Command & Control (C&C) servers.

Legitimate use-cases, such as enabling a covert communication for journalists, are also imaginable using network information hiding. However, most reported cases are criminal ones (Mazurczyk and Wendzel, 2018). In particular, network information hiding (and other forms of information hiding) were applied in several recent malware cases as we reported in (Cabaj et al., 2018a), Table 2.1. The Europol EC3-supported CUING initiative (*Criminal*

Malware/exploit kit	Information hiding method	Purpose
Vawtrak/Neverquest	Modification of the least-significant bits (LSBs) of favicons	Hiding URL to download a configuration file
Zbot	Appending data at the end of a JPG file	Hiding configuration data
Lurk/Stegoloader	Modification of the LSBs of BMP/PNG files	Hiding encrypted URL for downloading additional malware components
AdGholas	Data hiding in images, text, and HTML code	Hiding encrypted malicious JavaScript code
Android/Twitoor.A	Impersonating a pornography player or an MMS app	Tricking users into installing malicious apps and spreading infection
Fakem RAT	Mimicking MSN and Yahoo Messenger or HTTP conversation traffic	Hiding command and control (C&C) traffic
Carbanak/Anunak	Abusing Google cloud based services	Hiding C&C traffic
SpyNote Trojan	Impersonating Netflix app	Tricking users into installing malicious app to gain access to confidential data
TeslaCrypt	Data hiding in HTML comments tag of the HTTP 404 error message page	Embedding C&C commands
Cerber	Image steganography	Embedding malicious executable
SyncCrypt	Image steganography	Embedding core components of ransomware
Stegano/Astrum	Modifying the color space of the used PNG image	Hiding malicious code within banner ads
DNSChanger	Modification of the LSBs of PNG files	Hiding malware AES encryption key
Sundown	Hiding data in white PNG files	Exfiltrating user data and hiding exploit code delivered to victims

TABLE 2.1: Examples of existing information-hiding malware.

Use of Information Hiding), cf. (Mazurczyk and Wendzel, 2018), continuously monitors upcoming malware that uses information hiding methods.

2.2 Classification of Information Hiding in Communication Networks

In communication networks there is a rich diversity of opportunities for information hiding which can be called "localizations" or hidden data carriers due to their increasing complexity and sophistication. Therefore a great variety of information hiding techniques are potentially applicable. From the communications perspective three main types of information that can be subjected to hiding in networks (Figure 2.1):

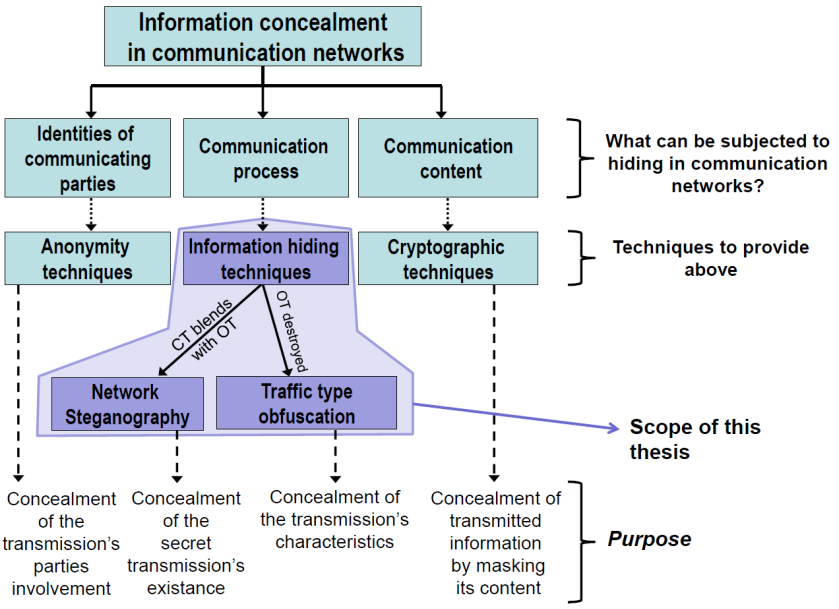

FIGURE 2.1: Classification of information concealment possibilities in communication networks

- **Identities of communication parties**, hiding of sender and/or receiver identities to ensure their privacy, which can be achieved with a variety of anonymity techniques (Danezis and Diaz, 2008).
- **Communication process**, concealing the existence of the fact that data exchange is taking place can be accomplished using network steganography techniques, while traffic type obfuscation is used to conceal the characteristics of the transmission.
- **Communication content**, cryptography-based techniques like encryption protect messages from disclosure to unauthorized parties thus concealing the content of exchanged messages. Other techniques in this group include, e.g., scrambling of the user's content.

From the three groups of information concealment methods for communication networks provided above, only the techniques that conceal the existence of the fact that data exchange is taking place are covered in this thesis (Figure 2.1). These methods can be further divided into *network steganography* and *traffic type obfuscation*, when we consider how they affect the overt transmission. Network steganography hides data inside an overt communication in a way that minimizes the impact on the overt transmission(s) and thus it effectively conceals the existence of the covert transmission (CT).

On the other hand, traffic obfuscation significantly changes or even completely alters an overt transmission (OT) to conceal the transmission's traffic characteristics by imitating another protocol's properties, for example by imitating the other protocol's packet frequency and/or size distributions. Both groups of techniques will be described in detail in the next sections.

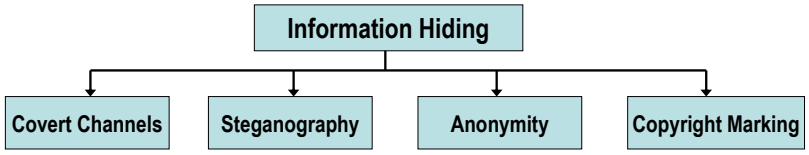

FIGURE 2.2: A historic classification of information hiding techniques (Petitcolas et al., 1999)

2.3 Evolution of Information Hiding Terminology

Steganographic methods are the oldest information hiding techniques, dating back as far as ancient Greece (Petitcolas et al., 1999). Steganography, a word of Greek origin meaning "concealed writing", encompasses all concealing techniques that embed a secret message (steganogram) into a carrier in such a way that the carrier modification caused by the embedding of the steganogram must not be "noticeable" to anyone. The carrier is suitable for steganographic purposes if it is commonly used. The form of the carrier has evolved over time, historical carriers were wax tablets, human skin or letters (Petitcolas et al., 1999), but these days it is digital pictures, audio, text or network protocols instead.

The last 15 years saw a very intensive research effort related to modern information hiding techniques. This was motivated by the industry business interested in DRM (Digital Rights Management) and the alleged utilization of steganographic methods by terrorists while planning the attacks carried out in the USA on the 11th of September, 2001 (Sieberg, 2001). The latter was not a standalone case as in 2010 it was reported that a Russian spy ring had used digital picture steganography to leak classified information from the USA to Moscow (Shachtman, 2010).

In 1996 at the first Information Hiding (IH) Workshop held in Cambridge, United Kingdom, the following information hiding classification was agreed upon as presented in Figure 2.2 (Petitcolas et al., 1999). It is worth noting that before this event the term "information hiding" was formally used to describe a computer programming technique – encapsulation that was introduced by Parnas in 1972 (Parnas, 1972).

In the classification presented by Petitcolas et al. (Petitcolas et al., 1999) the purpose of *steganography* was formulated as having a covert communication between two parties whose existence is unknown to a possible attacker; a successful attack consists of detecting the existence of this communication (the definition is consistent with the definition from Simmons "prisoners' problem" (Simmons, 1984)). *Copyright marking* deals with the embedding of marks into products. Copyright marks do not always need to be imperceptible. As opposed to steganography, copyright marking has the additional requirement of robustness against possible removal or alteration attacks as well as a fast embedding strategy in case of *transaction watermarking* (Steinebach et al., 2007). *Covert channels* were defined (supposedly after Lampson's definition (Lampson, 1973)) as communication channels that were neither designed nor intended to transfer information at all. Covert

channels are not necessarily invisible by definition but when established by steganography methods, covert channels are set-up in a way that they appear invisible nevertheless. In the remainder, we consider only such covert channels. *Anonymity* encompasses techniques that hide the identity of the communicating parties thus ensuring their privacy.

The classification in (Petitcolas et al., 1999) is semantically inconsistent and slightly misleading: (1) providing anonymity of communicating parties can be achieved by hiding information that identifies the communicating parties, but also, e.g., by clever routing of data using services such as *Tor* (Dingledine et al., 2004); (2) the distinction between steganography (and network protocol steganography in particular) and "covert channels" is not well grounded.

Obviously, the classification and definitions provided above were formulated considering the state of the art at the time the IH workshop was held. However, in the last decade the information hiding field evolved very dynamically and therefore further clarifications and improvements are required. The developments in the scientific community over the last 23 years must be taken into account. Information hiding techniques have evolved, and this should be reflected by refining their classification and definitions. Providing coherent definitions and classifications that reflect the current state-of-the-art in information hiding is not an easy task, as for more than a decade there has been an ongoing debate between scientists in this research field. One of the important issues is the discussion of whether the relationship between steganography and covert channels exist, and whether both terms are synonyms or not.

In our opinion, the current distinction between steganography and covert channels, when used to describe information hiding in communication networks, is artificial and even misleading. Both terms do not describe separate hiding techniques, and their current distinction is simply due to the evolution of the hidden-data carrier. However, as we shall see both terms are related to each other.

The scientific community has used different terms, such as steganography, covert channels, or information hiding, to describe the process of concealing information in a digital environment. The variety of terms comes from the fact that the different terms had not been introduced at the same time and because their definitions evolved. Investigating the evolution of the definitions of "covert channel" and "steganography" will help to demonstrate that, currently, the distinction between both terms is artificial, especially in a communication networks environment. In communication networks the hiding methods described by the two terms, *steganography* and *covert channels*, should be unified under a single term: *network steganography*. This does not mean that we want to discard the term covert channel. It is our view that network steganography techniques, as other steganography techniques, create covert (steganographic) channels for hidden communication, but such covert channels do not exist in communication networks without steganography (only the *possibility* for such channels exists a priori).

There are two widely known definitions of covert channels. Lampson (Lampson, 1973), who coined the term in 1973, defined a covert channel as "channel, (...) not intended for information transfer at all". The U.S. Department of Defense (DoD) modified Lampson's definition to emphasize the malicious intentions with which a covert channel can be utilized in so called "Orange Book" in 1985 (Department of Defense, 1985). The DoD defined a covert channel as "any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy" (thus implying certain applications of the covert channel). It is worth noting that both of these definitions do not imply what techniques are utilized to create the hidden communication channel. In addition, such policy-breaking channels are not even required to be *hidden* per definition. However, in the network steganography context, covert channels are almost entirely those that are considered as being hidden.

During the last two deades the covert channel's definition blurred and currently it is often used to describe particular information hiding methods, e.g., the hiding of information in unused/optional fields, rather than the associated communication channels. Moreover, in many papers (e.g. in (Petitcolas et al., 1999) and (Zander et al., 2007a)) steganography is intentionally distinguished from covert channels. Covert channels, as their name implies, are in general used for communication, but in theory one could also use them to store information, e.g., in network traffic traces or other network-related log files stored on hard disk. Steganography may be used to store information, e.g., one can embed secret data into a digital image and store it on a hard disk to hide the secret information from everyone, but of course it is also often used for communication. In fact, ancient steganographic techniques were often used for communication, i.e., hidden messages were concealed in innocent looking carriers to convey information to a receiving party aware of the steganographic procedure.

For example, let us consider again a famous steganographic method described by Herodotus (Petitcolas et al., 1999) where a trusted slave's head was shaved and then tattooed with a secret message about the planned revolt against the Persians. After his hair had regrown the slave was sent with this message to the Ionian city of Miletus. The goal of this steganographic method as well as many others that followed was to conceal the hidden communication by hiding secret information (in form of a tattoo) in a cover (carrier) of this data (tattooed human skin). Thus, the main aim of the historical steganographic techniques was to create covert channels (in uni- or bidirectional manner). In general, they were used not to store information concealed from everyone but to hide the information transfer.

In our opinion the main difference between the terms steganography and covert channels is merely the type of the carrier (cover) used; for steganography it is digital media like images, audio and video files whereas for covert channels it is a network protocol. Both terms exist because the communication methods used by people evolved from messengers, letters, and telephones to communication networks and accordingly the steganographic techniques evolved with the available communication methods. However, information hiding techniques that utilize network protocols should be seen as an evolutionary step of the carrier rather than some new phenomenon.

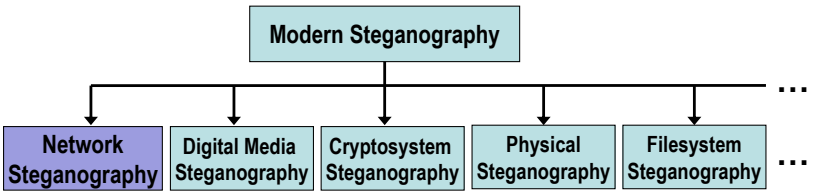

FIGURE 2.3: Classification of modern steganography techniques and scope of network steganography.

We propose to broaden the definition of the term covert channel so it describes any covert channel created by steganographic techniques, and we propose to use the term network steganography to describe the *methods* used for creating covert channels in communication networks. Our approach has the advantage that it provides the long-needed clarification but at the same time it is backwards-compatible with the existing literature. While the term network steganography should be used to describe a hiding technique in communication networks, it is not wrong to refer to the resulting covert channel.

2.4 Definitions, Classification and Characteristic Features

We propose the following classification of modern steganography techniques (not limited to communication networks) as presented in Figure 2.3.

The naming convention is *carrier-based*. This means that if network traffic is used as a carrier (the majority of so-called covert channels (Zander et al., 2007a)) then this is network steganography. When digital content, like images, audio or video files, is used as carrier then this is image, audio or video steganography (Bender et al., 1996a). In Figure 2.3 these techniques are aggregated into a single digital media steganography category. The previously named subliminal channels are renamed to cryptosystem steganography to ensure consistency. Other types of steganography mentioned in Figure 2.3 include, but are not limited to, physical steganography, where the most notable example is the previously mentioned information hiding by tattooing human skin, and filesystem steganography, where a filesystem's features are exploited for secret data concealment.

The main aim of network steganography is to hide secret data in the normal transmissions of users without significantly altering the carrier used. The scope of network steganography is limited to all information hiding techniques that:

- can be applied in communication networks to hide the exchange of data by creating covert communication channel(s).
- are inseparably bound to the transmission process (carrier).
- do not significantly alter the carrier.

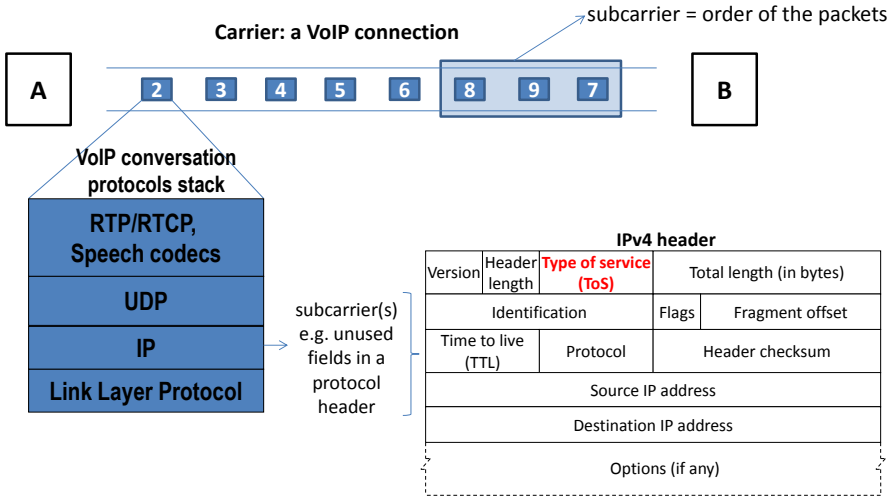

FIGURE 2.4: An example of carrier and subcarriers based on VoIP connection example (Mazurczyk et al., 2016b).

It must be emphasized that the main difference between "classic" steganography and network steganography is that the first relied on fooling human senses and the latter mainly tries to deceive other network devices (intermediate network nodes or end systems). For a third-party observer who is not aware of the steganographic procedure, the exchange of steganograms remains hidden. This is possible because the embedding of hidden data into a chosen carrier is "invisible" for parties not involved in the steganographic communication. Thus, the secret data is hidden inside the carrier, and the fact that the secret data is exchanged is also concealed.

In network steganography a *carrier* is one or more overt traffic flows that pass between a covert/steganogram sender and a covert/steganogram receiver(s). A carrier can be multi-dimensional, i.e., it offers many opportunities (places) for information hiding (called *subcarriers*). A *subcarrier* is defined as a "place" or a timing of "events" in a carrier (e.g. a header field, padding or an intended sequence of packets) where secret information can be hidden using a single steganographic technique (Figure 2.4). Typically, a subcarrier takes the form of a storage or a timing covert channel.

Subcarriers are also referred to as "cover protocols" (Wendzel and Keller, 2012c) but the definition of cover protocols is limited to storage areas in network protocol headers and also limited to the context of transferring control information in network covert channels whereas the term "subcarrier" can be applied in a broader meaning (cf. "Glossary").

As visualized in Figure 2.4 on VoIP connection example, multiple subcarriers can be used simultaneously within a single overt network flow. Another example that illustrates the same concept is when the steganographic carrier is an HTTP flow with two subcarriers, one which embeds secret data into the HTTP User Agent field and another that embeds secret data into the IP ToS (Type of Service) field. Both subcarriers could also be located within the same TCP/IP stack.

To hide secret data into a subcarrier it can be necessary to create space for its placement. For example, a covert sender can create a new IPv4 option or a new IPv6 destination option and then embed the secret data into this allocated space.

The most favorable carriers for secret messages in communication networks must have two features:

- they should be popular, i.e., the use of such carriers should not be considered as an anomaly. The more such carriers are present in a network, the easier it is to mask the existence of hidden communication.
- modification of the carrier related to embedding of the steganogram should not be "visible" to the third party unaware of the steganographic procedure.

Steganography relies on three characteristics of communications in current communication networks. First, a communication channel is not perfect. Errors and network anomalies, such as corrupted, lost or reordered packets, are a natural phenomenon and, thus, it is possible to embed information by mimicking them. Second, most network protocols specify fields or messages that are not used in all situations. This "surplus" can be used for embedding of secret data, if this does not degrade the carrier. Third, not every protocol is completely defined and "semantic overloading" is possible. Most of the specifications permit some amount of freedom in the implementation, and this can be utilized for steganographic purposes, e.g., HTTP header fields can be lower or upper case and secret data can be encoded via manipulating the case. Designing steganography-free protocols is hard and often practically impossible without unreasonably limiting the functionality or extensibility of protocols. Hence, network steganography can be achieved even with the simplest of communication protocols. However, more complex protocols usually offer more opportunities for sophisticated information hiding methods.

Considering the above definitions of carrier and subcarrier it is also important to emphasize that network steganography can be applied to *single* or *multiple* traffic flows. The example in Figure 2.4 presents the concept of single-flow network steganography, while Figure 2.5 visualizes the concept of multiple-flows network steganography. In the latter case the hidden data bits are distributed among many network flows. For instance, Figure 2.5 shows how binary "1" and "0" can be encoded into the sequence of packets from three different TCP connections.

We propose the following network steganography classification illustrated in Figure 2.6. It consists of three levels. Firstly, network steganography can be divided to *timing* and *storage* methods based on how secret data is encoded into the carrier (which is consistent with previously introduced covert channels classification). Of course, by combining the features of both methods, hybrid solutions are also possible.

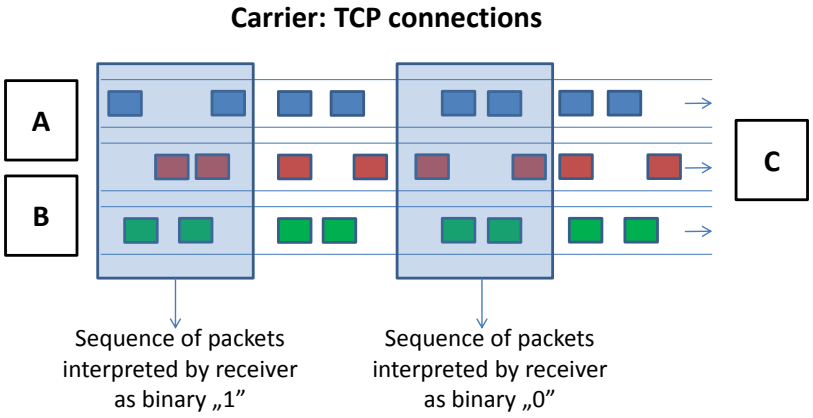

FIGURE 2.5: Multiple flows steganography example – sending secret data that is distributed over a number of traffic flows (Mazurczyk et al., 2016b).

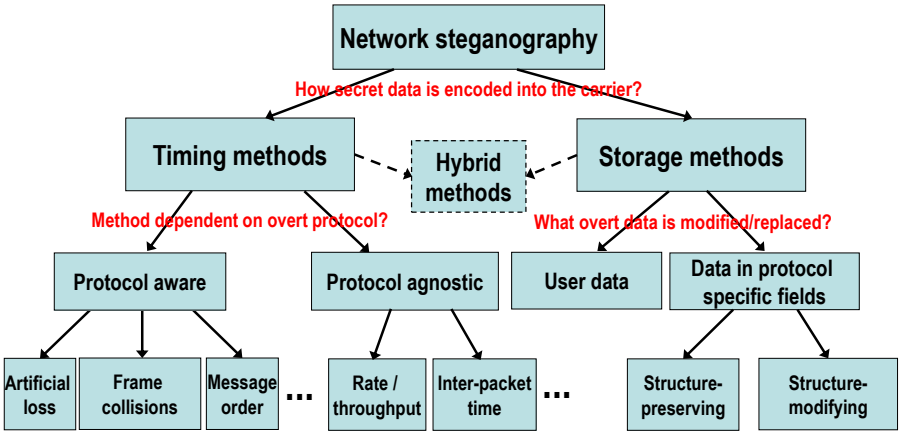

FIGURE 2.6: Network steganography methods classification.

Then each of two groups can be divided further. If we take into account whether timing method's depends on the underlying protocol we can distinguish between *protocol aware* and *protocol agnostic* methods. Protocol-aware timing methods require understanding of the carrier protocol – they utilize its characteristic features for hidden data exchange. For example, if a method exploits the fragmentation mechanism of a certain protocol, it needs to take into account how exactly this protocol's fragmentation mechanism works. On the other hand, protocol-agnostic techniques can be applied blindly to the selected carrier without in-depth understanding of the utilized protocol mechanisms and their specific features. For example, if one wants to encode secret data in inter-arrival time or data rate then a detailed knowledge of the carrier protocol is not required. However, even a protocol-agnostic method may still require knowledge of the traffic characteristics of the carrier protocol, so that a covert channel is created that looks like normal traffic.

Network steganography storage methods can be also split further based on what overt data is modified (which subcarrier is influenced). We can distinguish two groups: methods that *modify user data* and methods that *modify data in protocol specific fields*. The latter can be further divided into

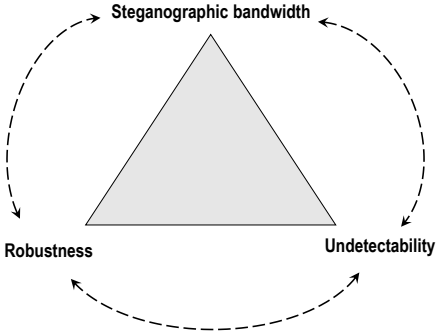

FIGURE 2.7: Relationship between the three features of network steganography (Mazurczyk, 2013).

methods that modify or replace existing protocol data and those that insert additional non-mandatory protocol data.

Each network steganography method can be characterized by three features. First, *steganographic bandwidth* describes how much secret data one is able to send per time unit. Second, *undetectability* is defined as an inability to detect a steganogram inside a carrier. The most popular way to detect a steganogram is to analyze statistical properties of the captured data and compare them to the typical properties of that carrier. Third, *robustness* is defined as the amount of alteration a steganogram can withstand without destroying the secret data. For each network steganography method, there is always a trade-off between maximizing steganographic bandwidth and still remaining undetected (and retaining an acceptable level of robustness). A user can utilize a method naively and send as much secret data as possible, but it simultaneously raises the risk of disclosure. Therefore, he/she must purposely limit the steganographic bandwidth in order to avoid detection. The relationships between these three features are typically described as magic triangle as proposed by Fridrich (Fridrich, 1999) and illustrated in Figure 2.7.[1]

Often it is also useful to calculate a *network steganographic cost* (Mazurczyk et al., 2016b). It describes the degree of degradation of the carrier caused by the steganogram insertion. The steganographic cost depends on the type of the carrier utilized, and if it becomes excessive, it leads to easier detection of the steganographic method. This characteristic indicates the degradation or distortion of the carrier caused by the application of a steganographic method. In digital media steganography and digital watermarking, i.e., for hiding secret data or for embedding a copyright mark in a digital image, audio, or video file, different parameters, such as JND (Just Noticeable Distortion), MSE (Mean-Square Error), PSNR (Peak Signal-to-Noise Ratio) are typically utilized for this purpose. However, these parameters cannot

[1] In other domains of information hiding, the triangle's features can be different. For instance, in *digital watermarking*, *undetectability* would be replaced with *transparency* while *steganographic bandwidth* would be replaced with *capacity*.

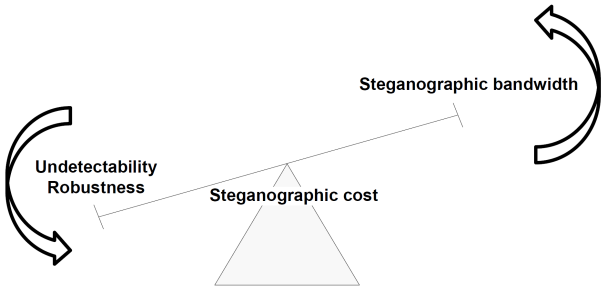

FIGURE 2.8: Relationship between the features of network steganography with steganographic cost included (Mazurczyk et al., 2016b).

be applied to dynamic, diverse carriers like network connections. For example, if the method uses voice packets as a carrier for steganographic purposes in IP telephony, then the steganographic cost can be expressed in conversation degradation. If the carrier is a certain protocol field, then the cost can be expressed as a potential loss in protocol functionality, etc. It is also possible that an information hiding method introduces steganographic cost that can be experienced in two different "planes", e.g., it introduces voice quality degradation as well as it adds additional delays to the overt traffic. If we include steganographic cost as a feature of a network steganography method, then the relationships with remaining features is as illustrated in Figure 2.8. When a steganographer wants to maximize steganographic bandwidth it typically results in increased steganographic cost, higher detetctability and lower robustness. Similarly, increasing robustness (e.g., using parity bits) can increase the steganographic cost. Therefore, in general, it can be concluded that the steganographic cost is linked detectability and robustness and may be responsible for the reduction of carrier's functionality or the reduction of carrier's performance (e.g. it results in increased transmission times or increased usage of resources).

The relationship between steganographic cost and detectability is explained in Figure 2.9. One can imagine a steganographic cost as a "zip" as it provides a view on how exactly the carrier was affected by applying a steganographic method. On the other hand, undetectability can be imagined as an "on/off switch". For a steganalysis method when a given level of steganographic cost (SC_T) is exceeded, the steganographic method becomes detectable with a probability greater than 50% ("flip a coin" chance of detection) up to the point where the detection is trivial ($SC_{D=100\%}$).

2.5 Traffic Type Obfuscation: Definitions, Classification and Characteristic Features

Traffic type obfuscation hides the nature of network traffic flows by obfuscating the underlying network protocol. That is, traffic type obfuscation modifies the patterns and contents of network traffic between two network entities so that a third entity is not able to reliably identify the type of their communication, i.e., the network protocol. We classify the existing work on traffic type obfuscation into two broad categories: *traffic de-identification*,

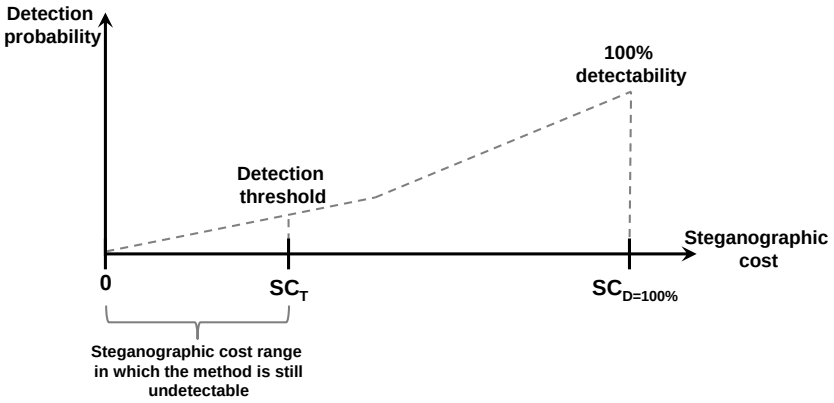

FIGURE 2.9: Relationship between steganographic cost and undetectability (Mazurczyk et al., 2016b).

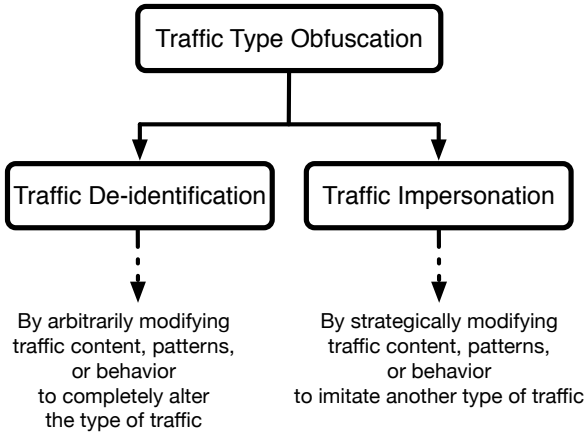

FIGURE 2.10: Traffic type obfuscation techniques classification.

and *traffic impersonation*. Traffic de-identification manipulates network traffic so that the underlying network protocol is hidden. For instance, network packet headers are 'encrypted' to conceal protocol identifier contents, thus obfuscating the underlying network protocol. On the other hand, traffic impersonation manipulates traffic so that it not only hides the underlying network protocol, but also pretends to be using another protocol, e.g., a target protocol. For instance, a BitTorrent client may alter its traffic to look like it is carrying HTTP traffic.

There are different reasons why someone wants to conceal network protocol through traffic obfuscation. First, it can be used to bypass network firewalls that disallow certain network protocols. For instance, many ISPs filter out peer-to-peer file sharing protocols like BitTorrent, where an efficient evasion is to obfuscate the protocol through traffic type obfuscation. Second, it can be used to conceal the existence of a type of communication. For example, the users of censorship circumvention tools may need to hide their use of circumvention systems, like Tor (Dingledine et al., 2004); this requires them to hide the Tor protocol, e.g., by morphing their traffic to look like a legitimate network protocol (Moghaddam et al., 2012).

We classify all techniques as traffic type obfuscation that make the network protocol unidentifiable with respect to:

- contents of network packets,
- statistical patterns of network packets, such as packet timing and sizes,
- and, protocol behavior, such as dynamic reaction to certain events.

As noted above, traffic type obfuscation shares intrinsic similarities with network steganography. Consequently, most of the terminologies and models used for traffic obfuscation are borrowed from the older, more established area of network steganography. More specifically, traffic type obfuscation is similar to network steganography, described in previous section, in that both aim to hide the existence of a covert communication. However, they differ in two main ways. First, in network steganography the overt channel is not significantly altered. By contrast, in traffic type obfuscation the contents of the carrier are “destroyed” by replacing it with the covert content. As a result, traffic type obfuscation techniques are able to provide higher capacities for covert communications. Second, traffic type obfuscation and network steganography differ in the application scenarios that they can be used for. Traffic type obfuscation can only be used when the carrier flow is generated by covert parties (covert senders and receivers). However, network steganography can be applied on traffic flows regardless of who has generated them.

As described before, the main objective of network steganography is to communicate the highest amount of secret information without being detected. In traffic type obfuscation, however, the objective is to conceal the type of the network traffic. As a result, the features such as "bandwidth", "robustness", and "cost" are not applicable to traffic type obfuscation. Instead, a type obfuscation scheme should be "unobservable": third parties should not be able to 1) detect that the traffic has been obfuscated, and 2) determine the original type of network traffic. The feature unobservability is equivalent to the feature "undetectability" used in the network steganography context.

2.6 Hidden Communication Model and Communication Scenarios

The state of the art communication model for steganography (Petitcolas et al., 1999) and covert channels (Zander et al., 2007a) is the same model. It is the famous "prisoners' problem", which was first formulated by Simmons (Simmons, 1984) in 1983 (Figure 2.11). In this model Alice and Bob are two prisoners that are trying to prepare an escape plan. The problem is that their communication is monitored by a Warden. If the Warden identifies any conspiracy, he will put Alice and Bob into solitary confinement making an escape impossible. So Alice and Bob must find a way to exchange hidden messages for their escape plan to succeed. The solution is to use steganography. By concealing the hidden message (M_{HID}) in an innocent looking carrier (M_{CAR}) it is possible to generate a modified carrier (M_{STEG}) that

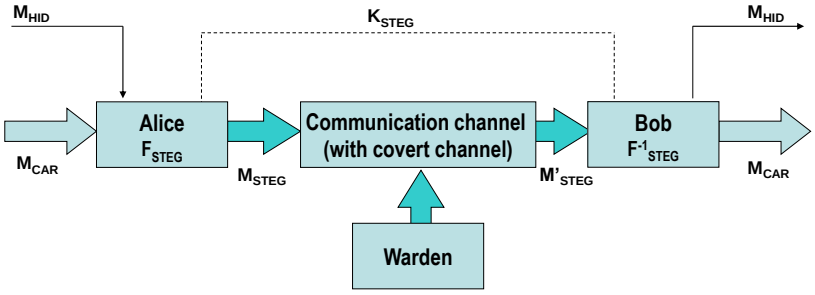

FIGURE 2.11: Model for hidden communication (Mazurczyk, 2013).

will raise no suspicion while traversing through the communication channel. For Alice and Bob the communication channel is also a covert channel that was created using a steganographic method.

Therefore, if there is no difference in the communication model it could be concluded that the term covert channel was used as another term for steganography (this fact was also observed by other researchers in the field, e.g., by Fridrich in (Fridrich, 1999)). However, we are convinced that the term covert channel cannot be used as a synonym for the term steganography. We believe that the term covert channel should be used to describe the communication channel that is created with the use of a steganographic method and this is consistent with Lampson's original definition of covert channels (see above). Thus, to be able to create a covert channel through which hidden data is exchanged, the sender (F_{STEG} in Figure 2.11) and the receiver (F^{-1}_{STEG} in Figure 2.11) must always utilize a steganographic method.

For steganographic methods it is usually assumed that there exists a secret stego-key (K_{STEG} in Figure 2.11) that is a form of shared secret between Alice and Bob. In network steganography a knowledge of *how the information is hidden* is the stego-key. In many cases network steganography achieves security through obscurity; only if a steganographic technique is unknown to the Warden, it can be used to exchange hidden data securely. However, some techniques are secure even if their existence is known, as long as the steganographic parameters are unknown to the Warden (e.g. properly encoded TCP ISN steganography (Murdoch and Lewis, 2005)). Hence, the stego-key is a combination of the steganography technique and its parameters used by Alice and Bob.

Apart from the stego-key everything else can be known to the warden. In particular, the Warden is aware that Alice and Bob can utilize a hidden communication, *can* know all other existing steganographic methods (unrelated to the stego-key), and can try to detect and/or interrupt the hidden communication. We refer to a *passive* Warden if the Warden tries not to eliminate but to detect a steganographic communication or if the warden tries to determine the involvement of a party into the steganographic communication. An *active* Warden, on the other hand, tries to eliminate or manipulate the steganographic communication. A *malicious* Warden can additionally alter Alice's or Bob's messages or even introduce fake messages into the communication (Craver, 1998).

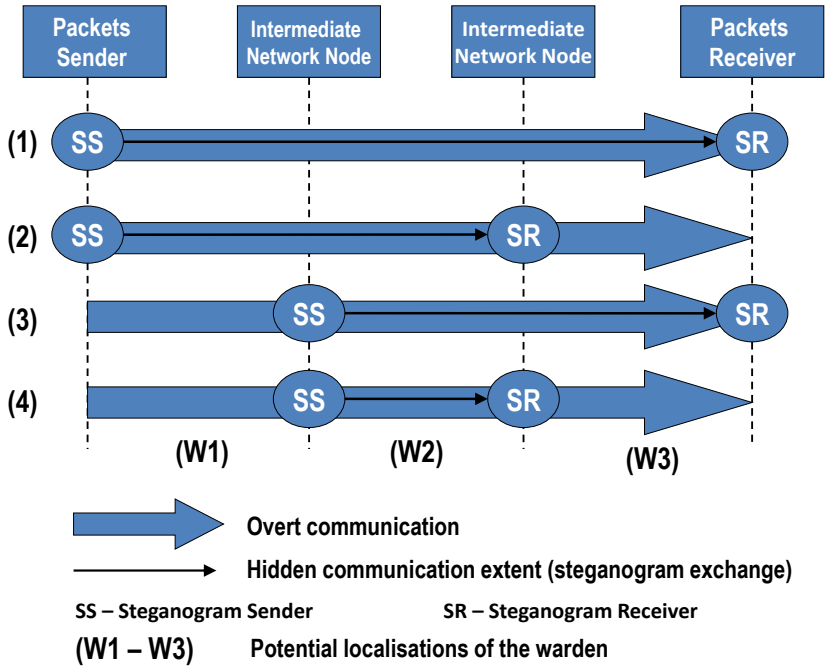

FIGURE 2.12: Hidden communication scenarios and potential localizations of the warden (Mazurczyk, 2013).

Let us consider the possible hidden communication scenarios (S1-S4 in Figure 2.12), as they greatly influence the detection possibilities for the warden. In general, there are three possible localizations for a warden (denoted in Figure 2.12 as W1-W3). A node that performs steganalysis can be placed near the sender, or receiver of the overt communication or at some intermediate node. Moreover, the warden can monitor network traffic in single (*centralized* warden) or multiple locations (*distributed* warden). In general, the localization and number of locations in which the warden is able to inspect traffic influences the warden's effectiveness.

The communication model is slightly different in the case of traffic type obfuscation. Unlike network steganography, which aims to send covert data, traffic type obfuscation tries to conceal the type of network traffic. Consider two entities Alice and Bob who aim to communicate in the presences of a warden. Suppose that the warden forbids any communication of type T1. Suppose that Alice and Bob want to communicate using traffic of type T1. To be able to do so, they need to obfuscate the type of traffic, e.g., by mimicking an allowed type of traffic, T2.

Moreover, unlike network steganography, traffic type obfuscation can only be applied when the traffic flows are generated by the covert entities. That is, traffic type obfuscation entities only obfuscate the network flows generated by themselves. As a result, the only communication scenario used in type obfuscation is the model (1) from Figure 2.12.

2.7 Information Hiding Countermeasures Models

Over the last few decades a number of countermeasures were developed against different network steganography and traffic obfuscation techniques. Here we provide an overview of the available countermeasures.

Before any action can be taken against an information hiding technique, it first needs to be identified, i.e., the Warden needs to become aware of its existence. Several formal methods were developed for identifying possible covert channels in specifications or implementations of operating systems or applications during the design phase or in an already deployed system (c.f. (P. R. Gallagher, 1993)). There also exist a few formal techniques for identifying possible covert channels in network protocols, many of which are adaptations of the mechanisms mentioned in (P. R. Gallagher, 1993) to network protocols (c.f. (Donaldson et al., 1988)). However, to the best of our knowledge none of these has been commonly used in practice.

Once an information hiding technique has been identified, the generally available countermeasures are:

- Eliminate the use of the covert channel.
- Limit the bandwidth of the covert channel.
- Detect and audit the covert channel.
- Document the existence of the covert channel.

If a known possibility for a covert channel was not removed or cannot be removed in the protocol design[2] or implementation, the next best option is to *eliminate the use* of the channel, because even low-capacity channels could be successfully exploited. The direct approach is to block or eliminate network protocols that enable network steganography. However, this approach is often impractical, since there are too many possibilities for hiding information in network protocols. The most common approach to prevent the use of network steganographic techniques and eliminate the resulting covert channels is *protocol normalization*.

Traffic normalizers (e.g. Snort (Snort Project, 2012)) remove semantic ambiguities by normalizing protocol header fields or timing behaviors of network traffic passing through the normalizer. For example, a normalizer can always set unused, reserved and padding bits to zero, thus removing the possibility to hide information in these bits. Traffic normalization can only be applied to all network traffic ('blind' normalization) if the normalization is transparent (it does not affect the traffic significantly). In practice, the use of normalizers may result in unwanted side-effects, since the normalization of header fields often means setting header fields to default values, i.e., these fields are not usable anymore. If accurate detection methods exist, detected covert channels can be eliminated using targeted normalization or even disruptive measures, e.g., the carrier traffic could simply be blocked.

[2]Some potential (low-capacity) covert channels must be accepted in every network protocol; no protocols are known to the author which are considered entirely free of covert channels.

The removal of all covert channels leads to very inefficient systems, since it typically means replacing automated procedures with manual procedures (Proctor and Neumann, 1992). Furthermore, covert channels based on the modulation of message characteristics are inherent in distributed systems, such as computer networks. Therefore, we and many other experts in the field (e.g., (Moskowitz and Kang, 1994)), believe that covert channels cannot all be completely eliminated. This is also acknowledged by the security standards. For example, the Orange Book treats covert channels with capacities of less than one bit per second as acceptable in many scenarios (Department of Defense, 1985).

If a channel cannot be eliminated its *capacity should be reduced* below an acceptable limit. What is an acceptable capacity depends on the amount of information leakage that is critical. For example, if the covert channel capacity is so small that classified information cannot be leaked before it is outdated, then the channel is tolerable. Limiting the channel capacity is often problematic in reality, because it means slowing down protocol mechanisms or introducing noise, which both limit the performance of the protocol. However, some limitation techniques showed good results in the literature as well as in practice, especially in the case of network steganographic timing methods that are hard to prevent. For example, the PUMP method (Kang and Moskowitz, 1993) is very effective in limiting (practically eliminating) timing channels in the flow of acknowledgment messages from a receiver to a sender needed for a reliable data transfer.

Covert channels that *cannot be eliminated or limited should be audited*, which requires reliable detection methods. Auditing acts as deterrence to possible users and also allows taking actions against actual users. (If the traffic data used for auditing can be obtained before traffic normalization takes place, one may want to audit the use of all covert channels.) The detection of covert channels is commonly based on statistical approaches or machine learning (ML) techniques (Zander et al., 2007a). In both cases the behavior of actual observed network traffic is compared against known assumed behavior of 'normal' traffic and (if known) the assumed behavior of carrier traffic with covert channels. The behavior is measured in the form of characteristics (also called *features* in ML terminology) computed for the actual observed traffic and traffic previously used to determine a *decision threshold* (or *decision boundary*) between normal traffic and traffic with steganography (training of the detection system).

With statistical approaches the decision boundary usually must be determined manually (by a human) during the training, and for some approaches proposed in the literature is not very clear how to choose an effective boundary. In contrast, ML methods determine the decision boundary automatically during the training phase. A disadvantage with some ML techniques is that the decision boundary can be very hard to interpret – effectively making the detection system a black box.

Supervised ML techniques require training with examples for both *classes* (normal traffic and traffic with covert channels). *Unsupervised ML* techniques or *anomaly detection* methods are usually trained with examples of normal traffic only. Supervised ML techniques are often more accurate, but also are less robust. Without proper retraining they can easily fail if

the characteristics of one of the classes change, e.g., due to a change of the covert channel encoding or a change of the normal protocol operations. Unsupervised techniques are often less accurate but are more robust against changes in traffic characteristics. With all detection methods it is important to avoid creating a detector that performs very well for the training data but does not perform well for the actual observed traffic (*overfitting* on the training data).

All known covert channels, except channels with capacities that are too low to be significant, should at least be documented (e.g. in the specification of network protocols). This makes everybody aware of their existence and potential threat, and it also deters potential users, since many steganographic techniques only provide security by obscurity.

As described before, the main objective of traffic type obfuscation is to conceal the type of network traffic, i.e., the network protocol being used. Consequently, any countermeasures against traffic type obfuscation will have one or both of the following two objectives: First, to detect network flows that try to hide their actual network protocols, and, second, to identify the actual network protocols of those network flows. If the actual protocol can be identified further actions can be taken in accordance with existing security policies, e.g., the obfuscated traffic can be blocked.

Countermeasures against traffic type obfuscation can be classified into three categories: content-based countermeasures (Houmansadr et al., 2013), which look for content discrepancies in the obfuscated traffic, pattern-based countermeasures (Dyer et al., 2012; Cai et al., 2012), which use statistical tools to identify obfuscated traffic based on their communication patterns (e.g., packet sizes and timings), and protocol-based countermeasures (Geddes et al., 2013; Houmansadr et al., 2013) that analyze the protocol behavior of traffic, e.g., by investigating their reaction to certain network conditions.

2.8 Outlook

Firstly, we expect that the application of network information hiding techniques leads to more sophisticated malware. This means that malware will be stealthier and thus harder to detect than today, supporting the advancement of Advanced Persistent Threats (APT). By deploying dynamic overlay routing and adaptive network covert channel techniques, malware will be harder to prevent on the network level. A host-based detection of malware is thus important when a network-level detection and prevention is not feasible in these situations. Today's command and control channels in botnets already possess a comprehensive feature set and increasingly adapting features to the context of network steganography is only one step in the malware evolution.

Secondly, we observe the increasing stealthiness of malware communications on smartphones. While there is not a major effort in developing novel network covert channels especially crafted for smartphones, recent trends take advantage on the device offloading features, especially those using the cloud. In fact, to bypass some storage or battery limitation of devices,

some operations are delegated to a remote server farm. In this perspective, new applications producing traffic potentially exploitable for network steganography are becoming available. Examples that we can expect to be exploited are voice-based services like Google Now and Siri or cloud storage platforms like Google Drive and Dropbox. Even if many frameworks apply protocols/techniques already used for network steganography (e.g., HTTP), the huge volumes and the degree of sophistication of many services will represent a challenge, especially in terms of being able to detect the covert communication or to provide effective countermeasures.

Thirdly, we expect the increasing emergence of network steganography to new domains, especially when combined with existing malware. One example in this regard is the potential to form novel botnets consisting of smart devices, allowing the remote mass-surveillance and remote control of the devices (Wendzel et al., 2014a). Network steganography can increase the stealthiness of mass surveillance in such situations, especially when the number of bots in a botnet is high.

Fourth, network steganography could increase the stealthiness of illegal data exchange, including communications we find within the darknet already today, such as the buying processes for the exchange of drugs.

Fifth, it can be expected that network steganography will influence industrial espionage when it comes to data leakage. While today's data leakage is performed quite plainly in many cases – e.g., data is leaked via email or USB stick – recent news have shown that steganography is indeed applied to leak data out of organizational environments. Using network steganography, data leakage cannot only be realized in a stealthy but also in a constant manner, e.g. by intentionally leaking a small amount of data per hour.

2.9 Conclusion

This chapter laid the foundation for a unified understanding of the network information hiding terminology and was a result of excessive discussions within the scientific community. It addressed fundamental differences between the network covert channel, network steganography and traffic obfuscation research communities so that previously used terms could be unified under the umbrella of a common discipline called *network information hiding*.

In particular, this chapter discussed information concealment possibilities in communication networks, dividing potential methods into hiding of communication parties' identities, the communication process they are involved in or the data that they are exchanging. However, the main focus was on techniques that enable covert communication, i.e., network steganography/covert channels and traffic type obfuscation. A key difference between these is how the overt communication is treated. In network steganography/covert channel research, the overt traffic's modification is very limited but in traffic type obfuscation the overt traffic is altered more extensively.

For both of these sub-disciplines, we characterized their main effectiveness features, communication models and typical usage scenarios. Finally, covert channel countermeasures were briefly discussed.

Chapter 3

Motivating Advances of Scientific Methodology in Network Steganography

Abstract After the previous chapter introduced improvements on the fundamental terminology in network information hiding, this chapter further motivates the methodological advancement of network steganography (and network covert channels). In particular, this chapter analyzes whether (and how) the research methodology in network steganography could potentially benefit from ideas that are linked to Science 2.0. Science 2.0 aims at using the information sharing and collaboration features of the Internet to offer new features to the research community. It has been already applied to computer science disciplines, especially bioinformatics. For network steganography, the application of Science 2.0 is still a rather uncovered territory. To foster the discussion of potential benefits for network steganography, we provide a disquisition for six different Science 2.0 aspects when applied to this domain.

Originally published:
Wendzel, Caviglione, Mazurczyk, and Lalande (2017b, *)

* The original publication was modified and extended for this thesis.

3.1 Introduction

This chapter uses the Science 2.0 paradigm as its core element. In short, Science 2.0 aims at fully exploiting the Internet to enable researchers to collaborate and share information (e.g., ideas, experiments, datasets and scientific papers) in order to increase both the volume and the quality of results, while mitigating costs. In other words, Science 2.0 is a driver for the digital transformation of science. Franzen argues that the digital transformation of science is irreversible (Franzen, 2018). Such an idea is not completely new, as it is an evolution of historical initiatives such as the Seti@Home project (Korpela et al., 2001).

As today, one of the most successful examples of Science 2.0 is given by myExperiment (Roure et al., 2009), which is a social website enabling to share scholarly information and scientific workflows in the field of bioinformatics. Another popular attempt is Galaxy Zoo (Zooniverse, 2019) using crowdsourcing to foster the collaboration among scientists for the morphological classification of galaxies. For the case of network security, there are no Science 2.0 initiatives comparable with the aforementioned ones. The only notable exception is given by arXiv (Cornell University, 2015), a database of preprints of scientific papers, which contains the cs.CR category for "Cryptography and Security". Even if it was intended as a place to store works from different research fields, e.g., mathematics, statistics and physics as shown in Table 3.1, it is not uncommon to find results dealing with computer science or network security.

We focus on *network information hiding*, in particular *network steganography* research, i.e., a discipline of network security that tries to hide the exchange of information on the network and that also tries to detect such stealthy communications.[1] For instance, network steganography is becoming an increasingly popular technique for malware, which can remain stealthy for a long time by cloaking the flows of stolen information within licit network traffic.

When searching the 2015 arXiv pre-print's abstracts for information hiding-related keywords, only 8 papers contain the terms "network" and "steganography" (network steganography is a sub-discipline of network information hiding), 16 contain "steganography" (which is a term that includes network information hiding but also other terms of information hiding, such as hiding techniques for audio or video content), and 6 contain "covert" and "channel" (a term to describe a hidden communication channel).

On the other hand, Figure 3.1 shows the yearly publications per search term obtained from Google Scholar between 1996-2015. As the comparison of both sources (arXiv and Google Scholar) indicates, the number of related publications per year that were indexed by Google Scholar (that also indexes non-open publications) is significantly higher, indicating that arXiv is not equally covering network information hiding publications.

With regard to security, steganography definitely plays a role since it has been used to increase the stealthiness of many hazards, for instance Internet

[1] As also done for the remainder of this thesis, we exclude traffic watermarking and anonymity research from our focus.

Category	cs	cs.CR	math	q-bio	q-fin	physics	stat
# papers	16179	828	28753	1558	689	8719	2541

TABLE 3.1: Statitics for arXiv pre-prints for the year 2015

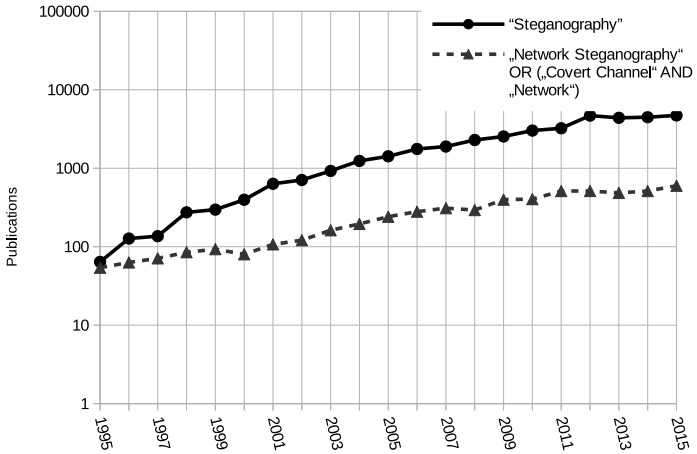

FIGURE 3.1: Google Scholar hits for selected search terms (1995-2015).

malware (Mazurczyk and Caviglione, 2015). In essence, it aims at studying techniques (and countermeasures) to inject secret data within an innocent looking carrier. It aims at cloaking a communication within network traffic to make any third part observers unaware of the undergoing data exchange. Moreover, even if important, it represents a small niche of network security. For this reason, the research community investigating network steganography is small and conferences explicitly dealing with such results are rare.

In this perspective, Science 2.0 could be an important enabler to help the network steganography community to reach a proper critical mass as well as to capture more attention from academics, vendors, and professionals working in the network security panorama. Therefore, the contributions of this chapter are: *i*) to propagate a tailored discussion of Science 2.0 in the area of network steganography in communication networks, *ii*) to foster the analysis of a Science 2.0-driven collaboration, experiment design, handling of re-inventions, tracking of progress in the research domain and potential impacts on teaching of network steganography, and *iii*) to enlighten possible matches among Science 2.0 and well-defined tasks that need to be undertaken by the research community dealing with network steganography.

3.2 Related Work

A large number of publications is available on scientific practice and Science 2.0 in the age of Web 2.0. For instance, Priem and Hemminger discuss scholarly impact metrics in the social Web, called "Scientometrics 2.0" in (Priem and Hemminger, 2010). The authors emphasize three important uses of Scientometrics 2.0, which are the evaluation of scholars, the recommendation of articles, and the study of science. In addition, the authors

mention several limitations of the paradigm, e.g., that Web 2.0 tools are replaced by other Web 2.0 tools in a frequent manner and can face spam.

Various advantages and drawbacks as well as the motivation of interviewed scientists when using Science 2.0 are highlighted in (Lin, 2012). Nattkemper discusses the use of Science 2.0 in applied (bio)informatics and medicine in (Nattkemper, 2012). He states that Science 2.0 is not used in its full capacity and solutions need to provide new data analysis methods for researchers to motivate a stronger value for them.

Franzoni and Sauermann (Franzoni and Sauermann, 2014) analyze an open collaborative fashion of scientific research which is often called Crowd Science. The authors identify two characteristic features of such crowd science projects, namely open participation and open sharing of intermediate inputs, and then explore crowd science's potential knowledge-related and motivational benefits. Bücheler and Sieg study the applicability of Crowdsourcing and Open Innovation in the scientific context (Bücheler and Sieg, 2011) while West *et al.* (West et al., 2014) review the contribution and evolution of Open Innovation from the history perspective. The includsion of Crowed Science and Crowdsourcing blurs the barrier between scientists and the population, which can bring new challenges with it (Franzen, 2018), e.g., the evaluation of scientific material and the quality of discoveries, also from the general population's perspective.

Alternative forms of online publications such as weblogs and *nano publications* are highlighted by (Franzen, 2018). She argues that such novel publication forms, including those with post-publication review allow the faster publication of (small) discoveries.

Laursen and Salter (Laursen and Salter, 2014) study a paradox of openness where firms (but also research teams) to innovate need to collaborate with many outside actors (research teams, organizations, institutions, etc.). Thus, the creation of innovations requires openness, but on the other hand the commercialization of innovations requires protection. Similar issues are often considered for Science 2.0. Anderson discusses several potential advantages of Science 2.0 in the conference review process for computer science (Anderson, 2009). We will refer to selected aspects of his work in the context of network steganography in Section 3.5.

3.3 Terminological Issues and Handling of Re-Inventions

A consistent terminology is essential for every domain to ensure the efficient progress of research work and the communication between scholars. In computer security and dependability, working groups such as the Fundamental Concepts and Terminology committee exist since decades to develop such a unified terminology (Aviẑienis et al., 2004).

So-called re-inventions have always been a component of science (Simonton, 2004). Especially due to the rapid development of different types of steganography methods and terminological inconsistencies in this field, various ideas how to hide data were invented multiple times (Wendzel et al.,

2015). The largest divergence in the network information hiding terminology can be found in the valley between the terms 'network covert channels' and 'network steganography' as these overlapping areas developed various similar/identical aspects. This thesis provides an approach to unify the terminology and categorization of hiding methods in the following chapters.

Currently, another problem is that many different steganography techniques can be utilized on a single device simultaneously. This is a consequence of the trend of devices combining features previously covered by several separate ones. For instance, smartphones offering a high-resolution camera, different air interfaces (e.g., Bluetooth, 3G and IEEE 802.11), and GPS. In this scenario, known classifications are too method-specific, thus, requiring a wider perspective. Especially, there is a need for a taxonomy allowing to grasp all the areas in which steganography can take place, thus, demanding for a 'cross-layer' scheme.

The need for the above-mentioned publications reflects the fact that terminological inconsistencies are present in the research domain. Science 2.0 tools can support a clean terminology and taxonomy. For instance, a collaborative Wiki can be used to discuss and merge terms. A similar approach, namely using a structured website with user comments for discussion of a research topic, is applied by the software patterns[2] research community, in which patterns are discussed online (Toxboe, 2019). Patterns can also be used to build terminological databases and taxonomies as they can form hierarchies. In network steganography, this is still not available and will be introduced in this thesis. In particular, the next chapters propose the creation of online platforms for pattern-based discussions in network steganography. By applying these research outcomes, unintentional re-inventions of hiding methods will not be eliminated but reduced.

On the contrary, we need to consider if there are points which speak against the use of Science 2.0 for improving terminology and handling re-inventions. Firstly, it could have only a moderate effect on the taxonomy of the research domain as existing publications feature old terminology and not all new research will adapt the new terminology. Secondly, online discussions can lead to *forks*, i.e., novel terminological paths that may improve the existing terminology but lead to even more terms. Thirdly, existing terminology and taxonomy working groups are already capable to achieve a high-quality output (e.g., (Avižienis et al., 2004)) without using Science 2.0 methods.

Thesis Context: The previous chapter improved the fundamental terminology for network information hiding, especially network steganography. During the next chapters, abovementioned patterns for hiding methods are presented. These patterns are introduced in Chapter 4 and provide an improved terminology and taxonomy for network steganography techniques.

[2]Software patterns are abstract descriptions of solutions to problems in a given context; they originate from the field of architecture. For instance, a pattern can describe a user-interface (solution) for a website (context) to achieve a suitable way to insert specific data (problem).

3.4 Experiments and Replications

Experiments in network steganography can be represented by two cases. Firstly, those experiments that test and measure the quantitative and qualitative aspects of hiding methods. Secondly, experiments that test and measure the quantitative and qualitative aspects of countermeasures for hiding methods. As reported in (Wendzel et al., 2015), more than hundred techniques for network covert channels are known. Also, a plethora of countermeasures are known for these techniques. For only few of these techniques, researchers can directly access and modify proof of concept implementations, experimental data, and exact workflow descriptions.

Collaborative Experiments Science 2.0 provides various online solutions, such as the mentioned *MyExperiment* (Roure et al., 2009) which allows the detailed description of particular scientific workflows. In network steganography, such workflows need to include information about the configuration of proof of concept codes, network interfaces, virtual machines, data to process, and all other components of the experimental design. The community can help to review experimental setups and can thus help to improve these before research work is actually submitted.

Collaborative tools such as *Github* (Github, 2019) allow the easy sharing of code and forking of software projects. Nevertheless, there is no effort for formalizing the set of inputs and to support the execution of experiments for a perfect reproducibility. New tools with less development functionalities but with better collaborative aspects for research appeared. For example, HubZero (McLennan and Kennell, 2010) allows to build a light virtual environment and to upload the software code of an experiment. Then, this environment acts as a module and can be called by other modules in order to benefit from the service. Each module, published under open source license, allows to produce new derivative experiments and obtain results using a physical back-end supporting the infrastructure. The platform reduces software and licensing costs for external researchers but (long-term) maintenance costs still remain for HubZero.

A variety of Science 2.0 tools to support experiments have been developed by physicists and biologists. These tools are barely used in information security, including network steganography. The reasons for non-use are: *i*) the lack of a common format for scientific data, *ii*) the lack of software components which can be run to conduct related experiments, and *iii*) the lack of detailed workflow descriptions which can be integrated into platforms such as *MyExperiment*. The adoption of Science 2.0 tools into the network steganography domain will be a clear advantage if these previous aspects can be improved by the research community.

In a further step, experimental setups could most easily be shared as virtual machines including pre-configured testbeds. Alternatively, research institutions with a strong focus on network steganography could provide a publicly accessible shared testbed for network steganography techniques to which every institution could contribute own (virtualized) machines which feature testbed setups. Such a research infrastructure will require massive

and long-term funding which could be a strong limitation. Possibly existing unused infrastructure could be utilized for such testbeds to reduce costs.

By releasing both covert channel tools and countermeasures tools under OSS licenses, e.g. as done in case of CCEAP (Wendzel and Mazurczyk, 2016), the research community could evaluate them in a comparable manner and experiments could easily be reproduced by third parties. Therefore Science 2.0 could form a common “battleground" to enable reliable comparative analysis of new and existing approaches for network hiding methods detection.

Legay Experimental Software Satyanarayanan reports about a project that archives legacy software to conserve historical scientific findings, including transcriptions from printed code to digital media (Satyanarayanan, 2018). Satyanarayanan moreover mentions related problems such as comparing legacy experimental results with results computed on modern hardware which, e.g., does not produce the same rounding errors during calculations. Another highlighted aspect is investigating false scientific results to determine whether calculation errors were caused by a researcher (e.g. entering the wrong formula in a spreadsheet tool) or by the legacy software itself that he or she has used.

The availability of legacy experimental environments and software would also be relevant to replicate experiments in network steganography, e.g. those conducted between the 1980’s and the mid-2000’s as well as younger experiments once they do not run on state-of-the-art computers anymore. However, due to the networked nature of the topic, it would become necessary to setup the network infrastructure in the same way, i.e., running the same old network cards and links (e.g., 10base2), outdated TCP/IP stacks including their bugs and so forth. While the general replication of an observed phenomenon might not require such old hardware, it would be required to falsify legacy experiments with certainty and in a way that it can be shown whether a scientist or a software/hardware caused false results. Even if old experiments do not directly apply to modern network environments, legacy results and replications could be useful on a meta-level. For instance, if an outdated network protocol has shown to be vulnerable to a certain type of covert channel, a study focusing on a currently used network protocol could benefit from the old results if both protocols show similarities in their specification (protocol design).

Thesis Context: This thesis contributes to the challenge of reproducing legacy experiments by introducing a way to unify the description of experiments (Chapter 7). These descriptions render experiments easier to compare by demanding certain properties to be described. They also aid the later replication.

Replication Studies *Replicability* in science refers to the capability of having different researchers conducting the same experiments as were originally conducted by some other researchers (Bajpai et al., 2019). (Glöckner et al., 2018) discuss problems related to replication studies in Psychology.

Their main points are that there is a need for an exact specification and revision of theories, a culture of transparent and acceptance of fallibility of revisions for empirical studies and theories; integration of such a culture in academic education; and establishment of joint and de-central databases for evidence-based and theoretical results. The authors moreover summarize the following fundamental problems based on related work: *i)* a lack of replicable experiments (only 36% of results are estimated to be replicable in Psychology; Psychology has established journals which link replication studies and replication projects), *ii)* the presentation of only selected experimental results, methods, analyses and studies, *iii)* incomplete documentation of experiments, gained data, questionnaires etc., *iv)* the post-hoc generation of hypotheses and explanations for obtained results, *v)* the use of sample sizes which are too small, *vi)* lack of independent falsification of results that leads to their publication, *vii)* explicit decisions not to publish results that are not conforming with previous findings (similar to point *ii*). Moreover, the authors desire to make science more *efficient*, that is optimizing the process of researching in a way that makes optimal use of resources (budget, hardware, staff) provided by the society. The authors report that even fundamental psychological effects that were published in papers with > 1000 and > 3000 citations could not be reproduced, leading to their removal from textbooks (Glöckner et al., 2018).[3] Moreover, the authors ask the question of when replication studies can be considered as valid and acceptable (this is especially depending on the original formulation of a hypothesis' conditions and discussed based on Popper's work on the logic of scientific discovery from 1934). Several solutions are proposed to improve the situation, e.g., making all research material and statistic findings publicly available through platforms such as *Curate Science* (`http://curatescience.org`), introducing the open science paradigm stronger into the university education and let students perform replications in their theses, or providing public funding for replication studies and replication databases/theory databases that link results. They moreover highlight upcoming problems, e.g., deciding who is allowed to decide whether a proposed replication study or theory can enter such a database.

All of the abovementioned problems are relevant to network steganography, resulting in very similar needs – be it the detailed description of experimental results, the provision of material, including code, and the fostering of a culture that values experimental replications. During reviews, code and dataset are often not provided, and experimental setups are lacking detailed descriptions. However, while the provision of such material might increase the efficiency of science in general, it is linked to a higher effort of the individual researchers as they need to provide additional material (code, documentation, datasets, ...) for a review process. Moreover, such a method requires reviewers to spend more time for review processes, which is unrealistic, given that the amount of papers reviewed by established scientists is already demanding. However, it could be considered

[3] Recently, discussions regarding the motivation of publishing fake results arose. Most researchers focus only on the validity of research as motivations are difficult to analyze. However, for brief coverage of potential motivations see (Schneier, 2019). Several hints for achieving replicability (and reproducibility) in computer networking research were summarized in the *Dagstuhl Beginners Guide to Reproducibility for Experimental Networking Research* (Bajpai et al., 2019).

feasible to demand improved experimental descriptions and provisions of code by ranked journals and conferences. In turn, this would decrease the number of papers to be reviewed (as more time would be required to prepare a submission and fewer papers would suffice the minimum requirements for submission) and would eventually allow a more detailed review.

Thesis Context: Problems *i* and *iii* of Glöckner et al. and – indirectly – *scientific efficiency* are addressed by this thesis in the following ways: the proposed pattern-based terminology and taxonomy (Chs. 2 and 4) as well as the review framework (Chapter 5) help to prevent scientific re-inventions as a unified understanding of terms (and their relations to each other) and a review process that explicitly checks for recurring ideas (patterns) counter the publication of redundant ideas under different terms. Moreover, patterns support *aliases*, which allow the inclusion of multiple previous terms under one pattern so that no existing ideas are left out and a reference to legacy terms is provided. Scientific re-inventions are costly and, thus, limit scientific efficiency. For this reason, the approaches of this thesis indirectly support scientific efficiency.
The introduced unified description method (Chapter 6) renders experiments comparable and serves as a fundamental element to support replications. With the WoDiCoF project (Chapter 7), this thesis additionally presents a testbed that can be used to conduct such replications. However, this thesis does not aim to provide a solution for deciding whether a replication study can be considered valid. WoDiCoF could be extended to serve as a platform comparable to Curate Science but it must be noted that Curate Science provides a much broader scope and an equivalent to Curate Science would be a platform for replication studies in the whole information security domain while WoDiCoF is limited to network steganography.

Competition and Overhead However, we also identified several difficulties for Science 2.0-driven experiments, especially for young researchers with a restricted budget.

Science 2.0 tools, especially in open communities, require individual scientists to share their insights, experimental setup details and tools. Providing these information means to give away an advantage, which is important for especially younger scientists who need to establish a unique profile in the community in order to reach a permanent position in academia.

The overhead of work to make experiments usable by other researchers could be too high for convincing researchers to publish their experimental systems (code, workflow descriptions, etc.). Processed data may be confidential, especially if captures of real network traffic are used, which may contain hidden messages from real attackers.

In addition, as Science 2.0 lowers the barrier for interacting with other research projects, it could become increasingly tempting for individual researchers to join a large number of research projects simultaneously. As stated by Bertolotti *et al.*, multi-team memberships of R&D teams do not solely lead to advantages but also to challenges for these teams (Bertolotti et al., 2015). For instance, if one researcher allocates little time on one of his many projects but some particular project demands his contribution, the progress made by the project can be decreased.

3.5 Tracking and Fostering Research Progress

The network steganography community is small in comparison to many other communities of information security. Its size may enable the manual tracking of the field's progress by a single researcher.

Tracking for Individuals Manual tracking consumes a larger percentage of the researcher's time. The support of Science 2.0 tools enables the faster tracking of the domain's developments. Science 2.0 tools are already used by a larger number of steganography researchers as the presence of online profiles reveals. However, as also reflected in (Van Noorden, 2014) the popularity of various Science 2.0 platforms differs significantly. For instance, in the field 'science and engineering' more than 90% of the researchers who reported to *Nature* said they are at least aware of *Google Scholar* (Google Inc., 2015) (or visit regularly) while less than 20% of the scholars were aware of *Microsoft Academic Search* (Van Noorden, 2014). A stronger use of the available tools will benefit the research domain. In addition, the scientific progress can be accelerated while allowing its tracking when Science 2.0 is used for the review process.

Fostering Research Progress Websites and apps such as Google Scholar, ResearchGate (ResearchGate, 2019), Overleaf (Overleaf, 2019), Authorea (Authorea, 2019), HubZero (McLennan and Kennell, 2010), PLOS (Public Library of Science, 2019) and Mendeley (Mendeley Ltd., 2015) provide reference managers, tools to read and annotate publications, and various ways for scientific collaboration. These platforms allow to track either particular researchers of a domain or a whole domain itself, for instance, by subscribing to search terms. Moreover, these platforms introduce evaluation metrics on the level of single articles and papers, e.g., PLOS' *Article Level Metrics* (ALMs) partially replace the classical journal *Impact Factor*) (Franzen, 2018). However, such metrics include "mentions" in journalistic online media, social networks or Wikipedia, rendering these article metrics dependent on public reception. For this reason, such metrics should not be considered a trustworthy value for assessing the scientific quality of contributions (Franzen, 2018).

Joint online writing and tools to perform experiments can help especially narrow fields to become more mature due to collaboration. In addition, such online collaboration tools increase the chances of individual authors for finding international co-authors. This factor is important as papers with multi-national authors increase the likeliness of citations (Khor and Yu, 2016), while citations and related research indicators influence research policies and performance incentives such as funding allocation (Kosten, 2016).

Another aspect of research progress tracking is the review process. Anderson suggests to improve the transparency of the review process in (Anderson, 2009). One of his suggestions is that all reviewed papers of a conference should be published online. Without Science 2.0, accepted papers appear in proceedings and journals while rejected paper will not appear although

they can still contribute significantly to research. For this reason, Anderson states that the *research community is worse off* if rejected papers remain unpublished (Anderson, 2009). This argument is supported by Mogul's statement that computer science reviewing is becoming *increasingly "hypercritical"* (Mogul, 2013). Additional evidence and discussion on this problem is provided by Meyer in (Meyer, 2011). Anderson mentions the fact that with the current system, some authors tend to submit unfinished work in the hope to get it accepted nevertheless. In network steganography, for instance, some submitted papers lack a strong evaluation or a proof of concept implementation. If even rejected submissions will be published online, authors may not want to see their names on their own publications as these are unfinished (Anderson, 2009). In this context, Science 2.0 can increase the quality of submissions by preventing intentionally low-qualified papers which are submitted only to receive review feedback. Instead of publishing unfinished work, ongoing work can be developed online together with other scientists before it will be submitted. Overall, such approaches, including the community review of web publications (Anderson, 2009; Mogul, 2013), could speed up the progress in steganography while also easing progress tracking.

However, novel science 2.0 review methods such as post-publication peer review or public peer review by *PubPeer* (The PubPeer Foundation, 2019) (and similar platforms) can shift the review process away from a purely scientific basis as they allow the inclusion of reviews conducted by non-scientists (Franzen, 2018).

Required Effort As already mentioned, Science 2.0 tools enable a faster tracking of the domain's progress. However, the field of network steganography may be too small for the utilization of *multiple* Science 2.0 tracking tools to be profitable for the individual researcher. In addition, the researchers' time to maintain online scientific data on several webpages or tools could require more time than the manual tracking itself. For instance, maintaining online research profiles requires the researcher to list his publications in each platform and to correct errors of automatically detected articles. Participation via Science 2.0 means to read and publish online discussions, to review peer's articles in various other platforms, and to perform collaboration using different tools at the same time. The combination of all these tasks can lead to a large overhead. Novel review forms such as public peer review exist but are linked to clear limitations, e.g., non-scientists performing reviews.[4]

[4]For this reason, the proposed pattern-based review process presented in this thesis is embedded into the traditional academic peer-review.

Thesis Context: This thesis fosters the academic peer-review process by the framework provided in Chapter 5 which aims to prevent scientific re-inventions and eases the peer-review process with a public database of hiding patterns. The framework can be used jointly with the unified description method that is proposed in Chapter 6 and which can be requested to be used by authors using reviewer comments. This way, the unified description method ensures that no essential aspects of a novel hiding method are left out in a scientific publication. Both contributions additionally ease the tracking of progress in the research domain.

3.6 Standardization

In network steganography there is a tight relation between the hiding method and carrier: the injection of hidden information primarily takes place by exploiting features of software implementations and protocol behavior with respect to the protocol's specification. In order to hide data within network traffic, precisely understanding how and where the needed information is stored within the related flow of packets is a mandatory step. At the same time, scientific literature already proved that there exists no general countermeasure to limit all forms of network steganography. For this reason, each hiding method must be carefully studied.

From Standards to Countermeasures By addressing steganography early during the design phase (e.g., by using formal methods such as the *Shared Resource Matrix* (SRM) (Kemmerer, 1983)) it should be possible to develop protocols more robust against data hiding. A tight interaction among researchers and developers could prevent issues, such as packets with unneeded fields or ambiguous meanings, which are a primary target for the injection of cloaked information. In this perspective, Science 2.0 would improve the pollination across academia and standardization bodies, which often neglect more theoretical and "what if" cases in favor of well-agreed implementations or practices. Thus, the collaborative nature of Science 2.0 could make the scientific pipeline (e.g., the process ranging from research planning to publication of results) more accessible. For instance, the standardization world can request and suggest ad-hoc tests, while the scientific world can perform state-of-the-art analysis on protocols albeit in an alpha phase.

For the case of Internet protocols, researchers and network/software engineers can collaborate through Science 2.0 tools in order to assess steganography risks early during the design stage. From the viewpoint of evaluating novel data hiding attacks, the access to the needed information is already possible by retrieving the proper *Request for Comments*, which are made publicly available by the *Internet Engineering Task Force* (IETF). One of IETF's founding rules is *rough consensus and running code*, what makes IETF a standardization body already compliant with the Science 2.0 paradigm. As consequence, this has two major implications: *i*) attackers and researchers have access to precise information about protocols and prototype implementations avoiding the need of performing reverse engineering or toy set-ups to

test the effectiveness of their network steganography methods, and *ii*) the standardization process is open to all participants, thus, enabling to early address steganographic threats during the development phase. In other words, when designing new protocols, addressing 'security' is mandatory for IETF.

In this perspective, the adoption of the Science 2.0 paradigm can boost the cooperation among the two worlds having positive impacts, such as: making the standardization community more aware of steganographic threats, establishing trial-and-error cooperation to mitigate the features that can be exploited for data hiding, and providing an unified and coherent knowledge to be used for the development on countermeasures. This is an important outcome as it can increase the chances of having countermeasures handled within the standardization pipeline and, eventually, increase their diffusion and adoption. Finally, pattern-based countermeasures could be developed to combat sets of hiding techniques with one logical approach. It is also imaginable to 'transform' one countermeasure to counter other covert channel techniques as originally desired.

Efforts for the Standardization Process Being able to actively participate in standardization activities, especially during the design phase of network protocols, requires both formal and technical knowledge. To this aim, Science 2.0 can improve the credibility of a (group of) scientist(s) but could fail to support claims to be pushed in the standardization pipeline. In fact, experimental set-ups are still only a starting point while standardization needs working prototypes and solutions rising a wide interest. Additionally, a standardization process requires the support of companies that have a strong interest to see the proposed standards adopted. These latter should have a major role in key sectors like telecommunications or the Internet of Things, where pushing countermeasures into new standards would have an impact. Using collaborative tools or open licenses may result in a conflict with these companies' policies and reduce the global effort in a standardization process where countermeasures would be seen as additional constraints for developing business.

Finally, transferring one countermeasure to work with another type of covert channel must be studied in detail for all types of covert channels, optimally in a pattern-based manner. Although this would require less effort than the development of new countermeasures for every single hiding technique, it will still be linked to a large amount of work with unclear outcome.

Thesis Context: This thesis addresses the problem of creating countermeasures for hiding techniques for which there is currently no known countermeasure. The so-called *countermeasure variation* is introduced in Chapter 8 and 'transforms' one countermeasure to a new pattern, i.e. it works on the basis of hiding patterns, and is experimentally evaluated for two specific countermeasures and multiple patterns.

3.7 Teaching in Higher Education

Only few courses on network steganography can be found at the undergraduate or graduate level (e.g., Master's level courses at the Warsaw University of Technology since 2012). Teaching is essential to keep the research domain alive and to allow its growth.

As mentioned in the previous chapter, network steganography methods are increasingly utilized for achieving hidden communication setups by various types of malware. This fact makes it important to incorporate lectures about steganography techniques, the threat they pose and possible countermeasures as an essential part of information/network security courses.

Designing Network Steganography Lectures Teaching of network steganography should be included at different levels of education including specialized courses for security professionals. The sooner the knowledge about recent advances in network steganography methods and countermeasures is disseminated among scientists, students and security professionals the higher the awareness and sensitivity for such threats. This is where Science 2.0 could play a significant role.

However, there are no unified methods to teach network steganography, especially when laboratory experiments must be performed (e.g., determining the capacity of a covert channel). As mentioned before, experimental setups and code are often not available. Hence, teaching cannot profit from it.

Teaching and Science 2.0-based Learning Social networks dedicated to self-learning tools like *Massive Open Online Courses* (MOOC) can be considered a core component of a Science 2.0-based learning. Using a creative common license to distribute courses supports the dissemination of materials and increases their visibility. Online courses could be created by cooperating groups that are specialized in network steganography. This would enable a detailed coverage of the most important aspects of this field. For the setup of testbeds with the already mentioned Science 2.0 solutions, such as myExperiment and HubZero, or tailored open source tools such as CCEAP (Wendzel and Mazurczyk, 2016), experimental setups can easily be made accessible to students and used for teaching purposes.

Nevertheless, ensuring the quality of online lectures and the easy application of available experiments in local setups can be difficult. Scientific experiments are designed in a detail level that is often too difficult to be used on an undergraduate level. Implementing hidden channels requires to know precisely the target programming language of the hosts (Java under Android, C for a regular Linux process, etc.) and to have advanced knowledge of networks. The amount of required knowledge and the programming skills could make it difficult to obtain an online course which is understandable, especially for MOOCs.

3.8 Required Effort for the Research Community

Compared to other research topics (e.g., network security intended as a monolithic area), the volume of works dealing with network steganography is quite modest, as also demonstrated by terminological issues (Wendzel et al., 2015). Therefore, the knowledge in terms of papers and prototypes needing to be migrated over Science 2.0 platforms could be easily handled by the research community. Figure 3.1 already highlighted the yearly publications per search term obtained from Google Scholar. Due to this still rather low number of publications, the effort to port past papers and research results into a Science 2.0 area could be feasible and could help potentially emerging network steganography into a Science 2.0-native discipline.

Nevertheless, groups performing research on network steganography appear as highly segmented. For instance, there are excellences studying the threat in smart buildings, mobile devices and in Voice over IP (VoIP) protocols. The migration of results towards a Science 2.0 approach requires cooperation and trust between the participating researchers.

These requirements could become a hurdle for achieving the cooperation of a larger number of research groups. A virtualized and segment-overlapping framework based on Science 2.0 would help research groups to have a greater critical mass, achieving a wider knowledge and develop more sophisticated methods and countermeasures. Moreover, papers may in future be authored by a larger number of researchers which are participating in the scientific process, resulting in lower career value for each author.

3.9 Conclusion

We highlighted six Science 2.0-related aspects and discussed their potential influence on network steganography. We see benefits of Science 2.0 for this research domain although the mentioned hurdles exist and several drawbacks must be considered. The size of the community, the inter-disciplinary nature of the field, the requirement of experimental setups and the links with standards and teaching facets result in a majority of the provided arguments. In particular, the bridge between reusable academic experiments and the effort in the standardization of countermeasures requires to have development and collaborative tools to structure the research community. For monitoring and disseminating the research results, Science 2.0 efforts are already ongoing.

Chapter 4

Hiding Patterns

Abstract After the first two chapters provided an improved fundamental terminology and taxonomy as well as a motivation for the methodological advancement in network steganography, this chapter will introduce the core element of this thesis: *hiding patterns*. Hiding patterns are applied on the level of hiding techniques and enable the methodological advancements of following chapters.

Within the last decades, various techniques for covert channels arose. We surveyed and analyzed 109 techniques developed between 1987 and 2013 and show that these techniques can be reduced to only 11 different patterns. Moreover, the majority (69.7%) of techniques can be categorized in only four different patterns, i.e. most of the techniques we surveyed are very similar. We represent the patterns in a hierarchical catalog using a pattern language. Our pattern catalog will serve as a base for future covert channel novelty evaluation. Furthermore, we apply the concept of pattern variations to network covert channels. With pattern variations, the context of a pattern can change. For example, a channel developed for IPv4 can automatically be adapted to other network protocols. We also propose the pattern-based covert channel optimizations pattern hopping and pattern combination. Finally, we lay the foundation for pattern-based countermeasures: While many current countermeasures were developed for specific channels, a pattern-oriented approach allows to apply one countermeasure to multiple channels. Hence, future countermeasure development can focus on patterns, and the development of real-world protection against covert channels is greatly simplified.

Originally published: Wendzel, Zander, Fechner, and Herdin (2015)

4.1 Introduction

As mentioned during the first two chapters, a large amount of research was accomplished within the last decades to evaluate attributes of network protocols (e.g. IPv4, IPv6, TCP, and HTTP) regarding their potential to hide information. On the other hand, only little work exists on providing a general, protocol-independent approach. Although coarse categorizations of network covert channel techniques exist (Zander et al., 2007a; Meadows and Moskowitz, 1996; Shen et al., 2005; Llamas et al., 2005; Zhiyong and Yong, 2009), no comprehensive and current catalog of the existing techniques is available.

Moreover, current techniques to counter network covert channels focus on single covert channels instead of common characteristics of multiple channels. The combination of dozens of countermeasures is required to achieve an acceptable protection, which is problematic in practice.

We consider a taxonomy for covert channel techniques very important in order to provide a framework to classify current and future research in the field, to determine similarities between techniques and to streamline the identification of novel countermeasures.

Patterns are a universal technique, which can be used to create taxonomies in a generic manner (Fincher et al., 2003). In particular, the *Pattern Language Markup Language* (PLML) provides a consistent formalization of pattern descriptions and is the standard pattern language in the human computer interaction field (Fincher et al., 2003).

We apply the approach of pattern languages to network covert channels, extract common patterns for hiding techniques and combine them in a novel hierarchy. In comparison to existing taxonomy approaches, we also cover covert channels from 2009 to 2013.[1] The focus of our pattern catalog is less on technical aspects but on the common abstract behavior of covert channel techniques, which is also a difference to existing categorizations.

We describe the identified covert channel patterns using an extensible PLML-based pattern catalog. Our catalog simplifies the future classification and novelty evaluation of upcoming covert channels. Only hiding techniques which require the integration of a new pattern into the catalog are very novel, others are simply variations of existing patterns. We show that the surveyed techniques can be reduced to only 11 different patterns. Moreover, the majority (69.7%) of techniques can be categorized in only four different patterns, i.e. most of the covert channel techniques we surveyed are very similar. Furthermore, our pattern catalog represents a systematic approach for identifying network covert channels in protocols in order to overcome the problem of requiring an *exhaustive search* (Sadeghi et al., 2012).

In addition, we present the idea of pattern *variation*. Pattern variation is based on pattern *transformation* (Engel et al., 2011; Engel et al., 2013), which allows authors and developers to alter the existing *context* of a pattern. For

[1] The original paper that serves as the basis for this chapter was submitted to ACM CSUR in 2014 and published in 2015. In follow-up papers, we evaluated more recent publications, cf. Chapter 6.

instance, a desktop browser interface pattern can be transformed to a user interface pattern for mobile devices and vice versa, i.e. the context changes from desktop to a mobile device (Engel et al., 2011).

Pattern variation is the first transformation-like approach for covert channels. We define the utilized network protocol as the pattern's context. Thus, a pattern's application can change from one network protocol to another – without re-implementing the hiding technique itself.

We also explain the improvement of pattern-based covert channels by introducing the concepts of *pattern combination* and *pattern hopping*. Pattern combination allows to use multiple patterns at the same time (e.g. for a single network packet or frame) to increase throughput while pattern hopping randomizes the use of patterns over time to increase stealthiness.

Furthermore, we motivate the development of countermeasures for network covert channels based on patterns. With patterns, covert channel protection in practice will become more realistic as the number of required countermeasures can be reduced greatly by targeting hiding techniques represented through generic patterns instead of aiming at specific hiding techniques.

The remainder of this chapter is structured as follows. Section 4.2 discusses previous taxonomies while Section 4.3 explains the concept of patterns and their use in our taxonomy. We introduce the identified covert channel patterns and our hierarchical pattern catalog in Section 4.4 and present our concept of pattern variation in Section 4.5. Section 4.6 motivates and discusses pattern-based countermeasures. A conclusion follows in Section 4.7.

4.2 Related Work on Covert Channel Classification

We now describe existing surveys and classifications of network covert channels, and how our novel pattern-based classification improves on these.

An early taxonomy of covert channels in multilevel security systems (MLS) was presented by Meadows and Moskowitz (Meadows and Moskowitz, 1996). Covert channels were associated with four different contexts based on the service conditions in which they occur: High-to-low service covert channels, low-to-high service covert channels, shared service covert channels, and incomparable service covert channels. The taxonomy in (Meadows and Moskowitz, 1996) concentrates on early covert channel techniques that break security policies but do not necessarily provide stealthy communication. Our work focuses on network covert channels and on their hiding techniques instead of the service conditions in which they appear.

Shen *et al.* classified local covert channels and proposed the idea to counter local covert channels based on their characteristics (Shen et al., 2005). In comparison, our work discusses network covert channels and provides a hierarchical and more extensive categorization.

Llamas *et al.* surveyed covert channels in Internet protocols (Llamas et al., 2005). The first part of their paper covers fundamentals of covert channel research for both, local and network covert channels. The second part

summarizes publications on network covert storage and timing channels in TCP/IP protocols. However, (Llamas et al., 2005) merely lists the different covert channels and makes no attempt to categorize them.

In 2007, Zander *et al.* (Zander et al., 2007a) published a comprehensive survey on network covert channels. The work covers the terminology, adversary scenario, covert channel techniques and countermeasures. Moreover, a categorization is applied which differs between channels taking advantage of unused header bits, header extensions and padding, the IP Identifier and the Fragment Offset, the TCP Initial Sequence Number (ISN), checksum fields, the Time to Live (TTL) field, the modulation of address fields and packet lengths, the modulation of timestamp fields, packet rate and timing, message sequence timing, packet loss and packet sorting, frame collisions, ad-hoc routing protocol-based techniques, Wireless LAN techniques, HTTP- and DNS-based techniques, application layer protocol-based channels and payload tunneling. The categorization in (Zander et al., 2007a) is fine-grained: channels are not only categorized by their underlying technique, but also by the protocol layer they operate on. Also, (Zander et al., 2007a) does not provide a hierarchy or a standardized pattern definition.

Zhiyong and Yong proposed a taxonomy based on entropy, and their work is the closest to our own (Zhiyong and Yong, 2009). Each channel falls into one out of three categories depending on the 'source' used to encode the covert data: variety entropy, constant entropy or fixed entropy. As in our own work, (Zhiyong and Yong, 2009) motivates the development of prevention techniques with a focus on covert channel categories instead of single techniques. We propose a hierarchical and more fine-grained categorization, which is not based on entropy but on the actual hiding techniques. While the development of *more detailed and particular* countermeasures for the provided classification was left for future work in (Zhiyong and Yong, 2009), our categorization enables more practical countermeasures that can address covert channel patterns.

We define an improved network covert channel taxonomy based on the pattern language PLML/1.1 and present the concepts of pattern variation, pattern combination and pattern hopping. Pattern *variation* allows the adaption of one hiding technique to arbitrary network protocols. Pattern *combination* allows to simultaneously use multiple patterns for a single PDU (e.g. a single network packet) or for a sequence of PDUs, while pattern *hopping* randomizes multiple pattern-based covert channels to improve stealthiness. Pattern hopping enables a channel to adapt itself to changing conditions in networks (e.g. switching to another pattern if one pattern is blocked). Another advantage over the existing surveys is the fact that we also include publications from 2009 to 2013 into our categorization. Besides presenting the patterns, we also discuss countermeasures in the context of the identified patterns.

4.3 Pattern-related Fundamentals and Their Taxonomy Use

After introducing fundamentals and related work on covert channels, we now cover the fundamentals of patterns and pattern languages. Moreover,we describe our use of PLML.

4.3.1 Patterns and Pattern Languages

Graphical notations like UML 2.0 are powerful modeling languages for the description of specifications and the subsequent documentation (Object Management Group (OMG), 2010) but only represent the end result of the design process. Since the late 70s *patterns*, in comparison, enable the successful documentation of design decisions during the development process. In 1977, Alexander introduced first design patterns for solving problems in architecture and urban planning (Alexander et al., 1977). Eighteen years later, the *Gang of Four* (GoF) transferred the pattern concept to the domains of software architecture and software engineering (Gamma et al., 1995). Today, patterns are also used in fields of human computer interaction (HCI) research (Fincher et al., 2003), usability engineering (Marcus, 2004), user experience (Tiedtke et al., 2005), task modeling (Gaffar et al., 2004), and application security (Yoder and Barcalow, 1997).

As shown in Figure 4.1, all patterns represent a relation between a certain design problem and a solution in a given context. The *problem* is a description of the issue to be solved. The *solution* refers to a specific design that solves the given problem. The *context* describes a repetitious set of cases in which the pattern can be used.

The use of patterns has a number of advantages (Seffah, 2010): patterns are simple and easily readable for designers, developers and researchers, and they are useful for the collaboration between the people involved. Furthermore, patterns are based on established knowledge and capture fundamental principles for good designs. Patterns also specify requirements in a general way that allows different implementations of the same pattern.

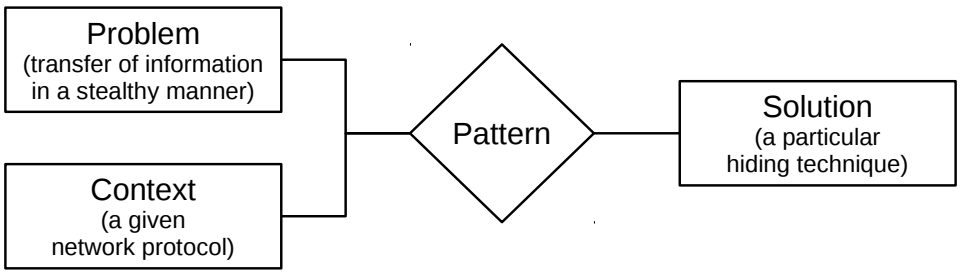

FIGURE 4.1: The concept of patterns

4.3.2 Utilization of PLML for a Covert Channel-based Taxonomy

In order to ensure a certain standard, patterns are summarized in a so-called *pattern catalog* (Alexander et al., 1977). A catalog of related patterns that belong to a common domain is a so-called *pattern language* (Seffah, 2010). Today, widely accepted pattern catalogs exist in the field of HCI, such as the catalogs presented by van Welie (Welie, 2001), Tidwell (Tidwell, 2009), and van Duyne (Van Duyne et al., 2007). However, these pattern catalogs are described in different styles.

To enable the clear definition and comparison of patterns, so-called *schemes* were introduced (Alexander et al., 1977). These schemes are divided into sections of textual and graphical descriptions. However, no standardized description of pattern scheme *attributes* existed, as numerous pattern catalogs were created, based on a different understanding of attributes. Due to these inconsistencies, searching and referencing patterns across different catalogs is difficult. To overcome this problem, a standardized pattern language was developed based on XML: the *Pattern Language Markup Language* (PLML) v. 1.1 (Fincher et al., 2003). PLML unifies and standardizes the schemes of different authors with the help of XML tags. Each XML tag represents a part of the scheme. Table 4.1 describes the tags from the PLML 1.1 we used to define covert channel patterns.

Tag	**Description**
<pattern id>	Identifies a pattern within the particular catalog.
<name>	A correct assignment of a name for each pattern is important for the retrieval of a pattern when the pattern becomes part of a second catalog.
<alias>	Patterns can have different names, which are specified in the <alias> tag. The alias tag helps to find the same pattern when the pattern has different names in different catalogs.
<illustration>	An application scenario for the pattern.
<context>	Specifies the situations to which the pattern can be applied.
<solution>	Describes the solution for a problem to which the pattern can be applied. The attributes *problem* and *context* (cf. Figure 4.1) are usually blurred but often not separated into two attributes.
<evidence>	Contains additional details about the pattern and its design. Moreover, the tag can contain examples for known uses of the pattern.
<literature>	Lists references to publications related to the pattern.
<implementation>	Introduces existing implementations, code fragments or implementational.

TABLE 4.1: Used PLML/1.1 Attributes

4.4 Classification of Covert Channel Patterns by using PLML

We evaluated the covert channel research of the last decades and classified the 109 evaluated covert channel techniques into 11 abstract patterns.

4.4.1 Coverage of Techniques

To select the most significant covert channel techniques for our pattern catalog, we took the existing surveys by Llamas *et al.* (Llamas et al., 2005), Zander *et al.* (Zander et al., 2007a) and Zhiyong and Yong (Zhiyong and Yong, 2009) into account and included the referenced publications in our evaluation. Moreover, we cover additional papers with a significant amount of citations or novelty that were not mentioned in the surveys (e.g. because of their later publication between 2009 and 2013).

4.4.2 Pattern List

We now describe all patterns. For better readability, in the following textual presentation of the pattern catalog we merged the content of the *literature* and *evidence* attributes in the *evidence* attribute and removed pattern attributes from PLML which are either redundant or not pertinent to our contribution. Some papers propose various techniques belonging to the same pattern (e.g. (Lucena et al., 2006) presents 10 hiding techniques forming part of the *Add Redundancy* pattern). In such cases, we do not mention all techniques explicitly.

For some patterns less than three use cases exist in the literature. In such cases, we added our own ideas for hiding techniques to provide a minimum of three use cases. Our hiding techniques have no citation, as they are initially proposed in this chapter.

P1. Size Modulation Pattern:

Illustration: The covert channel uses the size of a header element or of a PDU to encode the hidden message.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying

Evidence:

1. Modulation of data block length in LAN frames (Girling, 1987)
2. Modulation of padding field's size in IEEE 802.3 frames (Wolf, 1989)
3. Modulation of IP fragment sizes (Murdoch and Lewis, 2005; Mazurczyk and Szczypiorski, 2012)
4. Modulate the message length of network packets (Ji et al., 2009)
5. Modulate the size of IPSec messages (Sadeghi et al., 2012)

6. A man-in-the-middle adversary between VPN sites actively manipulates the maximum transmission unit (MTU) within the *path MTU discovery* process between the VPN sites. Path MTU discovery is a continous process and changed MTUs are propagated to systems within the VPN site, i.e. allow to encode hidden information within the MTU (Sadeghi et al., 2012).

P2. Sequence Pattern:

Illustration: The covert channel alters the sequence of header/PDU elements to encode hidden information.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying

Evidence:

1. Sequence of Hypertext Transfer Protocol (HTTP) header fields (Dyatlov and Castro, 2005)
2. Sequence of Dynamic Host Configuration Protocol (DHCP) options (Rios et al., 2012)
3. Sequence of File Transfer Protocol (FTP) commands (Zou et al., 2005)

P2.a. Position Pattern:

Illustration: The covert channel alters the position of a given header/PDU element to encode hidden information.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying → Sequence

Evidence:

1. Position of an IPv4 option in the options list of an IPv4 packet
2. Position of an IPv6 extension header in the list of extension headers
3. Position of a DHCP option in the options list (Rios et al., 2012)

P2.b. Number of Elements Pattern:

Illustration: The covert channel encodes hidden information by the number of header/PDU elements transferred.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying → Sequence

Evidence:

1. Alter the number of options placed in an IPv4 packet
2. Modulate the number of options placed in a DHCP packet (Rios et al., 2012)
3. Modulate the number of fragments created from an original IP packet (Mazurczyk and Szczypiorski, 2012)

P3. Add Redundancy Pattern:

Illustration: The covert channel creates new space within a given header element or within a PDU to hide data into.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying

Evidence:

1. Generation of packets with IPv4 options that embed hidden data (Trabelsi and Jawhar, 2010)
2. Create a new IPv6 destination option with embedded hidden data (Graf, 2003)
3. Extend HTTP headers with additional fields or extend values of existing fields (Dyatlov and Castro, 2005)
4. Manipulate the *pointer* and *length* values for the IPv4 record route option to create space for data hiding (Trabelsi and Jawhar, 2010)
5. Add random bytes to an encrypted SSH message (Lucena et al., 2004)
6. Extend Simple Mail Transfer Protocol (SMTP) packet headers with additional fields (Getchell, 2008)
7. Hide data in unused bits of the DHCP *chaddr* field if the *hlen* field is set to a value that is larger than the size of a network address (Rios et al., 2012)
8. Encapsulate IP packets with a smaller size than specified in the Ethernet frame size and use the space between the end of the IP packet and the Ethernet trailer for covert data (Muchene et al., 2013)
9. Encode hidden information through the presence/absence of "type" or "xml:lang" attributes in the Extensible Messaging and Presence Protocol (XMPP) or through the presence of leading/trailing white spaces in XMPP messages (Patuck and Hernandez-Castro, 2013)

P4. PDU Corruption/Loss Pattern:

Illustration: The covert channel generates corrupted PDUs that contain hidden data or actively utilizes packet loss to signal hidden information.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Modifying

Evidence:

1. Generate corrupted messages in broadcast erasure channels (Servetto and Vetterli, 2001)
2. Transfer corrupted frames in IEEE 802.11 (Kraetzer et al., 2006)
3. A man-in-the-middle adversary between two VPN sites drops selected packets exchanged between the VPN sites to introduce covert information into an established connection of adversaries located within the VPN sites (Sadeghi et al., 2012).

P5. Random Value Pattern:

Illustration: The covert channel embeds hidden data in a header element containing a "random" value.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Preserving → Modification of an Attribute

Evidence:

1. Utilize the IPv4 *Identifier* field (Rowland, 1997)
2. Utilize the first sequence number of a TCP connection – the Inital

Sequence Number (*ISN*) (Rowland, 1997; Rutkowska, 2004)
3. Hide data in the TCP ISN using a bounce server (Rowland, 1997)
4. Utilize the DHCP *xid* field (Rios et al., 2012)
5. Utilize the Secure Shell (SSH) protocol Message Authentication Code (*MAC* field) (Lucena et al., 2004)

Notes: As some header elements, such as the TCP ISN, follow a distribution which conforms to a particular operating system or context, their values cannot be considered perfectly random and the placement of "random" values in such elements can lead to different value distributions, which can be detected (Murdoch, 2007).

P6. Value Modulation Pattern:

Illustration: The covert channel selects one of n values a header element can contain to encode a hidden message.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Preserving → Modification of an Attribute

Evidence:
1. Send a frame to one of n available Ethernet addresses in the local network (Girling, 1987)
2. Encode information by n of the possible IP header Time-to-live (TTL) values (e.g. a high or a low TTL value) (Zander et al., 2006)
3. Encode information by n of the possible *Hop Limit* values in the IPv6 header (Lucena et al., 2006)
4. Encode information by sending a packet using one of n possible application layer protocols (Wendzel and Zander, 2012) or application layer ports (Borland, 2008)
5. Encode information by selecting one of n possible messages types in the Building Automation and Control Networking (BACnet) protocol (Wendzel et al., 2012)
6. Encode information in the target IP of address resolution protocol (ARP) messages (Ji et al., 2010)
7. Change the value of the "type" or "xml:lang" attributes in XMPP (Patuck and Hernandez-Castro, 2013)
8. Send IPSec packets from one VPN site to specific destination IPs within another VPN site (Sadeghi et al., 2012).

P6.a. Case Pattern:

Illustration: The covert channel uses case-modification of letters in header elements to encode hidden data.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Preserving → Modification of an Attribute → Value Modulation

Evidence:
1. Case modification in HTTP headers (Dyatlov and Castro, 2005)
2. Modify the case of the "type" or "id" attributes in XMPP (Patuck and Hernandez-Castro, 2013)

3. Case modification in SMTP, Post Office Protocol (POP3), or Network News Transfer Protocol (NNTP), commands and headers

P6.b. Least Significant Bit (LSB) Pattern:

Illustration: The covert channel uses the least significant bit(s) of header elements to encode hidden data.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Preserving → Modification of an Attribute → Value Modulation

Evidence:

1. Encode into the IPv4 timestamp option by effectively sending at even/odd times (Handel and Sandford, 1996)
2. Modify the low order bits of the timestamp option in TCP (Giffin et al., 2003)
3. Utilize the least significant bits of the *secs* field in the DHCP header (Rios et al., 2012)
4. Encode covert bits in slight modifications of view angles (yaw, pitch) of player's avatars in the Quake3 multiplayer game protocol (Zander et al., 2008)
5. Utilize the least significant bits of the IPv4 *TTL* field
6. Utilize the least significant bits of the IPv6 *Hop Limit* field (Lucena et al., 2006)
7. Utilize the least significant bits of the *Hop Count* field in the network layer PDU of the BACnet protocol
8. Utilize the least significant bits of the "id" attribute in XMPP (Patuck and Hernandez-Castro, 2013)

P7. Reserved/Unused Pattern:

Illustration: The covert channel encoded hidden data into a reserved or unused header/PDU element.

Context: Network Covert Storage Channels → Modification of Non-Payload → Structure Preserving → Modification of an Attribute

Evidence:

1. Utilize undefined/reserved bits in IEEE 802.5/data link layer frames (Wolf, 1989; Handel and Sandford, 1996)
2. Utilize unused fields in IPv4, e.g. Identifier field, Don't Fragment (DF) flag or the reserved flag, as well as in IP-IP encapsulation (Handel and Sandford, 1996; Ahsan and Kundur, 2002a; Buchanan and Llamas, 2004; Sadeghi et al., 2012)
3. Encode hidden data in unused or reserved fields of the IPv6 header or its extension headers ((Lucena et al., 2006) lists 8 hiding techniques for the Reserved/Unused pattern in IPv6)
4. Utilize unused bits in the TCP header (Handel and Sandford, 1996)
5. Utilize the ICMP echo payload (Stødle, 2011; daemon9, 1997)
6. Utilize the padding field of IEEE 802.3 (Wolf, 1989; Jankowski et al., 2010)

7. Utilize unused fields in the BACnet header (Wendzel et al., 2012)
8. Place hidden data behind the string termination symbol in the *sname* and *file* fields of DHCP (Rios et al., 2012)
9. Place hidden information into the *Differentiated Services* (DS) field of outbound IPSec connections (Sadeghi et al., 2012).
10. Insert hidden data into the IP *Explicit Congestion Notification* (ECN) field in IPSec connections (Sadeghi et al., 2012).

P8. Inter-arrival Time Pattern:

Illustration: The covert channel alters timing intervals between network PDUs (inter-arrival times) to encode hidden data.

Context: Network Covert Timing Channels

Evidence:

1. Alter the timings between LAN frames sent (Girling, 1987)
2. Alter the response time of a HTTP server (Esser, 2005)
3. Alter the timings between BACnet/IP packets (Wendzel et al., 2012)
4. Introduce artificial delays into inter-arrival times of SSH packets sent based on keyboard input (interactive shell) (Shah et al., 2006)
5. Acknowledge IEEE 802.2 I-format frames immediately or after a second I-format frame was received (Wolf, 1989)
6. A man-in-the-middle (MitM) adversary in the public network between two VPN-secured sites modifies the inter-arrival times of packets transferred between two man-in-the-edge (MitE) systems on each site of the VPN to signal hidden data to both MitE adversaries (Herzberg and Shulman, 2013).
7. Alternatively to (6), the MitE systems communicate covertly by sending traffic with manipulated inter-arrival times to each other (Sadeghi et al., 2012; Herzberg and Shulman, 2013).
8. Record a legitimate traffic sequence, partition the sequence and replay the inter-arrival times of a particular partition (Cabuk, 2006)

P9. Rate Pattern:

Illustration: The covert channel sender alters the data rate of a traffic flow from itself or a third party to the covert channel receiver.

Alias: Throughput Pattern

Context: Network Covert Timing Channels

Evidence:

1. Exhaust the performance of a switch to affect the throughput of a connection from a third party to a covert channel receiver over time (Li et al., 2011)
2. Manipulate the serial communication port's throughput by delaying *Clear to Send/Ready to Send* commands (Handel and Sandford, 1996)
3. Directly alter the data rate of a legitimate channel between a covert channel sender and receiver.

P10. PDU Order Pattern:

Illustration: The covert channel encodes data using a synthetic PDU order for a given number of PDUs flowing between covert sender and receiver.

Context: Network Covert Timing Channels

Evidence:

1. Modify the order of IPSec Authentication header (AH) packets (Ahsan and Kundur, 2002a)
2. Modify the order of IPSec Encapsulating Security Payload (ESP) packets (Ahsan and Kundur, 2002a)
3. Modify the order of TCP packets (Luo et al., 2007; El-Atawy and Al-Shaer, 2009).
4. A MitM adversary in the public network between two VPN-secured sites modifies the order of packets transferred between two MitE systems on each site of the VPN to signal hidden data to both MitE adversaries (Herzberg and Shulman, 2013).
5. Like (4), modify the order of IPSec packets for inbound or outbound VPN traffic (Sadeghi et al., 2012).
6. A covert channel sender transfers frames in a way they are sent before or after a legitimate user's frames in CSMA/CD networks. The covert channel receiver analyzes the order of arriving frames (Handel and Sandford, 1996).

P11. (Artificial) Re-Transmission Pattern:

Illustration: A covert channel re-transmits previously sent or received PDUs.

Context: Network Covert Timing Channels

Evidence:

1. Transfer selected DNS requests once/twice to encode a hidden bit per request.
2. Duplicate selected IEEE 802.11 packets (Kraetzer et al., 2006)
3. Encode hidden data by re-transmitting selected TCP segments.
4. Do not acknowledge received packets in order to force the covert sender to re-transmit a packet. The re-transmitted packet is modified by the sender to carry hidden data (Mazurczyk et al., 2011).

4.4.3 Taxonomy/Classification

We provide a hierarchical view of the discovered patterns in order to structure our findings. The hierarchy is visualized in Figure 4.2 where white boxes represent categories of patterns and gray boxes represent patterns. Conforming to the PLML standard, a covert channel pattern can also be a *child pattern* of a parent pattern, as in case of the *Case pattern*.

The major categorization of all network covert channels is into timing and storage channels. We introduce additional sub-categories for storage channels due to their diversity. We distinguish between storage channels which

apply hiding methods to payload (e.g. to audio streaming) – these channels are outside of our scope – and storage channels which alter non-payload (e.g. header elements or padding bits). These non-payload modifying channels do either change or preserve the structure of a PDU – a novel difference we discovered in the analysis process.

In case of a structure modification, a pattern either alters the order of elements in the protocol header or it changes the size of the PDU. We discovered different patterns for both variants. On the other hand, if a pattern preserves the structure of a PDU, the pattern must modify a data element in the PDU (e.g. a header field).

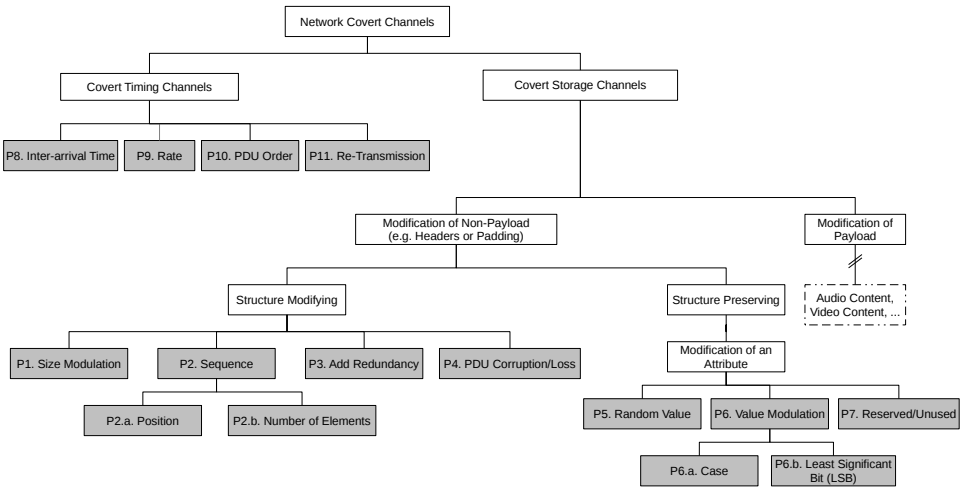

FIGURE 4.2: Network covert channel pattern hierarchy, excluding hiding techniques utilizing payload

Besides the given hierarchical representation, Table 4.2 categorizes all patterns regarding the following additional aspects:

- **Semantic:** The semantic of a PDU is changed if the pattern modifies header elements in a way that leads to a different interpretation of the PDU. For instance, the semantic of an IPv4 header is changed if a "record route" option is attached but preserved if the reserved flag is set as the reserved flag does not lead to a changed interpretation of the packet. In general, a channel raises less attention if the semantic of network data is not modified.
- **Syntax:** We call a modification of the PDU structure a syntax modification. For instance, adding additional header elements changes the PDU structure. As with the semantic, a covert channel pattern can either modify or preserve the syntax. The syntax categorization is only applied to storage channels as timing channels do not change the structure of a PDU.
- **Noise:** In general, all covert channels can be *noisy* since network frames or packets of the overt channels can be reordered, modified or lost, which can lead to bit errors or bit deletions/erasures in the covert channel. However, storage channels exploit the fact that overt protocols, such as TCP, have mechanisms for reliable data transport. If the header fields in which the covert channels are encoded are not

changed in the network, these channels are effectively *noise-free*. Timing channels on the other hand are always noisy, since the network always affects the timing of frames, packets or messages depending on the network conditions (e.g. congestion). Active wardens (e.g. traffic normalizers) may also introduce noise. However, we do not consider this type of noise as part of the covert channel characteristics.

In general, noise in the form of bit corruptions or packet loss can affect all presented patterns. Here we only categorize a pattern as noisy if the channel is a timing channel and thus, always faces noise, or if it is embedded in PDU fields which are modified in the network (like the TTL in the IPv4 header). Although active wardens can introduce additional noise in many patterns (e.g. by removing IPv4 options used to carry hidden data), we do not take normalization effects into account.

	Semantic		**Syntax (Structure)**		**Noise**	
	preserving	modifying	preserving	modifying	noisy	noiseless
Storage Channel Patterns						
P1. Size Modulation	X			X		X[2]
P2. Sequence	X[3]			X		X
P2.a. Position	X[4]			X		X
P2.b. Number of Elements	X			X		X
P3. Add Redundancy	X			X		X
P4. PDU Corruption/Loss	-[5]	-[6]		X	X	
P5. Random Value	X		X			X
P6. Value Modulation		X	X		X[7]	X[8]
P6.a. Case	X		X			X
P6.b. LSB		X	X		X	
P7. Reserved/Unused	X[9]	X[10]	X			X
Timing Channel Patterns						
P8. Inter-arrival Time	X		-	-	X	
P9. Rate	X		-	-	X	
P10. PDU Order	X[11]	X[12]	-	-	X	
P11. Re-Transmission	X		-	-	X	

TABLE 4.2: Categorization of Covert Channel Patterns

[2]Fragmentation can cause noise for channels using the Size Modulation pattern since routers can fragment large packets into multiple smaller packets.

[3]The semantic of Sequence and Position patterns is only preserved if an utilized element's position or the sequence of elements have no effect on the PDU's semantic.

[4]See previous note (3).

[5]Intentionally corrupted PDUs are not interpreted and thus do neither change nor preserve the semantic of a PDU.

[6]See previous note (5).

[7]A value modulation can lead to noise (e.g. if the IP TTL is used) but can also be noiseless (e.g. if the source address is modulated).

[8]See previous note (7).

[9]The semantic of a PDU *can* change if the covert channel modifies currently unused elements (e.g. a set DF flag in the IPv4 header would prevent fragmentation). On the other hand, a utilization of currently unused elements can *preserve* the semantic, e.g. if the covert channel sets the MF flag in IPv4 to zero, the modification of the Fragment Offset will not lead to a changed semantic.

[10]See previous note (9).

[11]If the channel utilizes a protocol that provides packet sorting at the receiver side, the PDU Order pattern can preserve the semantic of the data transfer, otherwise it can change the semantic.

[12]See previous note (11).

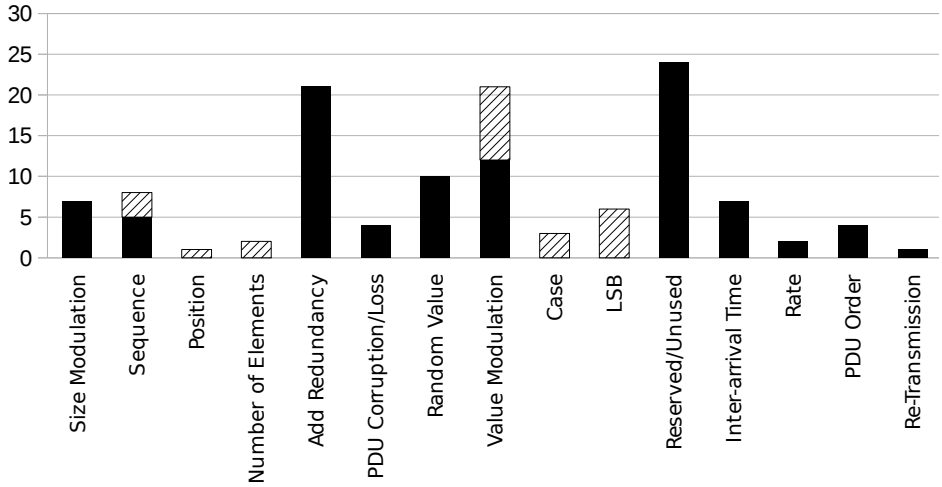

FIGURE 4.3: Number of associated covert channel techniques per covert channel pattern. Shaded bars represent child patterns.

4.4.4 Occurrence Rate of Particular Patterns

Besides the fact that we were able to 'reduce' the 109 hiding techniques to only 11 patterns, our findings also show that the majority (76 of 109 or 69.7%) of hiding techniques is based on only four patterns, namely the Reserved/Unused pattern (24 techniques), the Add Redundancy pattern (21 techniques), the Value Modulation pattern (21 techniques, including its child patterns Case and LSB), and the Random Value pattern (10 techniques). In other words, many of the surveyed and analyzed techniques are of relatively little novelty as they are based on existing techniques. Figure 4.3 compares the number of covert channel techniques associated with the particular patterns.

It would be interesting to compare the occurrence rate of patters in the literature with their actual number of uses in practice. However, no information about the usage rates of covert channels are available.

4.4.5 Extensibility of the Pattern Catalog

The patterns introduced in our work will be made available publicly online in a moderated wiki. A wiki allows the collaboration among experts from both, research institutions and industry: it allows active discussion and, through moderation, the controlled extension and modification of the pattern catalog.

If a novel hiding technique requires the integration of a new pattern, researchers can add the pattern to the catalog, which will be invisible until accepted by the moderators. The acceptance may be performed after detailed discussion with the contributing researchers to ensure the wiki's accurate and consistent state with the existing pattern collection and to prevent future contributors from wrongly classifying their "new" covert channel techniques as new patterns. Moreover, researchers can add upcoming

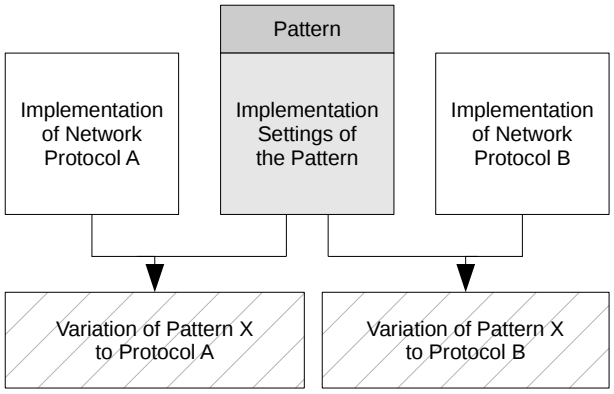

FIGURE 4.4: Concept of Covert Channel Pattern Variation

hiding techniques that are based on an already existing pattern to the *evidence* property of the particular pattern.

4.5 Variation of Covert Channel Patterns

Covert channels are not only used for malicious purposes (e.g. the control of botnets) but are also used to support the freedom of speech (e.g. of journalists). Furthermore, research on covert channel improvements enables the further development of necessary countermeasures against malicious users; the research community must identify improvements for covert channel techniques before malicious users take advantage of them.

We will now describe the variation of the patterns defined in the previous section. A pattern variation automatically adapts a pattern to a new context. In case of network covert channels, we define the *utilized network protocol*, used as carrier for the hidden data, as the *context*. If the channel switches the protocol, the context changes as well and thus, the pattern must be adapted to the new context. For instance, a pattern that was previously applied to hide data in IPv4 can be adjusted to hide data in IPv6.

A pattern variation is only useful if it is an automatic process for both, the sender and the receiver. The automatic process generates new code that implements the pattern for the altered context. Thus, a pattern variation eliminates the necessity of programming a pattern from scratch if it needs to be adapted to another network protocol.

Figure 4.4 visualizes the concept of a covert channel pattern variation: While a general network protocol implementation is required to hide data within a protocol or the protocol's timing behavior, a pattern has information which enable the application of the pattern to other network protocols.

4.5.1 PLML-based Pattern Variation

Our approach is similar to that of Engel *et al.* who store the information required for a pattern transformation within the *implementation* property of PLML (Engel et al., 2011; Engel et al., 2013). We introduce so called *settings*

```
settings.ipv4.Offset=32;
settings.ipv4.Len=16;
settings.tcp.Offset=32
settings.tcp.Len=32
settings.tcp.OnlyFirstPkt=true
```

FIGURE 4.5: Settings for IPv4 and TCP in case of the Random Value Pattern

into the implementation property. These settings contain all significant information for a variation.

Engel *et al.* define a configuration for each context of a pattern (Engel et al., 2013). We match this aspect as well by introducing specific settings for each supported overt network protocol. For instance, a protocol A may provide 4 bits of space for a pattern and the location of the bits is at an offset of 6 bits from the first bit of the header. In the new context of protocol B, the pattern can only use 3 bits and these 3 bits are located at an offset of 20 bits from the first bit of protocol B's header. In other words, each variation of a pattern highly depends on the utilized network protocol. Given a very simple hiding technique that just utilizes a single bit and sets it to "1" or "0" in an arbitrary protocol with a static header structure, the variation would just need an "offset" value for each protocol to locate the selected bit in its header.

Figure 4.5 shows example settings for the *Random Value* pattern. For IPv4, these settings would utilize the 16 bit Identifier field and for TCP, the 32 bit ISN would be used. As the ISN is only random in the first packet of a flow, a limitation (*OnlyFirstPkt*) must be enforced.

We found that the following settings can be necessary for covert channel patterns – depending on the particular pattern:

- *Offset*: Number of bits between the first bit of the protocol header and the first bit of the utilized area
- *Len*: Size of the utilized area
- *OnlyFirstPkt*: Only use the first packet of a flow (e.g. for the TCP ISN)
- *Min/MaxSize*: Minimum/maximum size of a (padding) field or of a frame
- *Min/MaxElements*: Minimum/maximum number of elements to use (e.g. minimum number of IPv4 options or DHCP options for the Position pattern or the Sequence pattern)
- *ValueRange/ValuesAllowed*: These fields limit the range of allowed values for a field and the particular values which can be stored in the field. For instance, the Value Modulation pattern may only be allowed to place the values "a"-"j" in a field or only the values "yes", "no" and "optional". For passive covert channels, i.e. those piggybacking third-party traffic, constraints can be defined in order to allow only limited modulations. For instance, a constraint can allow only TTL modifications in a range of +/-1.

```
settings.ipv4.value=0,scapystring=IP(ttl=100+RandInt()%50);
settings.ipv4.value=1,scapystring=IP(ttl=150+RandInt()%50);
settings.ipv6.value=0,scapystring=IPv6(hlim=100+RandInt()%50);
settings.ipv6.value=1,scapystring=IPv6(hlim=150+RandInt()%50);
```

FIGURE 4.6: Sample tuples using *scapy* strings for the LSB pattern

- *Min/Max/DistributionIPG*: Minimum/maximum time between packets or definition of a value distribution for these time differences. While this attribute is generally useful, it is especially required to configure limits for the Inter-arrival Time pattern.
- *Min/MaxRate*: Minimum/maximum packet rate. This attribute is similar to the Min/MaxIPG attribute but configures the number of PDUs the channel is allowed to transfer per time t. The attribute is of general use but especially required to configure the limits for the PDU Order pattern to ensure stealthiness. In future work rate profiles could be defined as well in order to match the actual traffic behavior as close as possible.

While the actual hiding technique of a pattern can be adapted automatically, protocols with header fields which depend on other header fields still require the additional use of libraries and tools like *scapy*[13]. For instance, the calculation of the IPv4 *Checksum* as well as the adjustment of the *Internet header length* and *total length* fields are not included in the pattern variation process and must be done additionally. We therefore decided to include *scapy* commands into our settings as *scapy* automatically calculates values for header fields the user does not explicitly define. Thus, the use of *scapy* eliminates the need to implement a variation's utilized network protocols. Therefore, additional settings must be defined in the form settings.**protocol**.value=**bit value**,scapystring=**command**.

Figure 4.6 shows a sample setup for assigning two bit values to two different protocols (IPv4 and IPv6) of the LSB pattern using *scapy* strings. A high and a low value are assigned to TTL or Hop Limit in order to transfer a "0" (*value=0*) or a "1" (*value=1*) bit. In order to prevent a trivial detection of the channel, the values are randomized in a higher/lower range.

Like our pattern catalog, the settings proposed for pattern variation also serve as a basis for future work, i.e. they can be extended by adding additional settings in the future.

4.5.2 Requirements-based Pattern Variation

Pattern variation is also applicable in the context of situational requirements. For instance, the transfer of a video stream over a covert channel requires a higher throughput than the transfer of a password within the same time.

Each overt channel (the utilized network protocol) provides a particular amount of space to carry hidden data and has a different potential to raise

[13] http://www.secdev.org/projects/scapy/

attention (Wendzel and Keller, 2011). A pattern variation for a given requirement can thus also switch the overt channel used in order to provide a high throughput or a high covertness. If the pattern is required to transfer a video stream it may use an overt channel providing a high throughput while the pattern may use an overt channel with a low throughput and high covertness if only a few hidden bits must be transferred. Therefore, a pattern cannot only change the overt channel but can also adjust the number of manipulated bits in the overt channel.

4.5.3 Similar Approaches to Pattern Variation

Covert channels can utilize multiple hiding techniques simultaneously, cf. (Wendzel and Keller, 2011; Yarochkin et al., 2008). For instance, one technique could modify the LSB of the IPv4 TTL while another technique modifies the IPv4 reserved flag.

This section puts *patterns* in the context of such a simultaneous application of hiding techniques. To this end, we describe two approaches that utilize multiple patterns instead of multiple hiding techniques, namely *pattern combination* and *pattern hopping*.

Pattern Combination

To increase the throughput of a covert channel and its stealthiness, we can combine multiple patterns, e.g. by applying them in parallel to a single packet – an idea already shown for single hiding techniques in the field of network steganography (Fraczek et al., 2012a) – or sequentially to subsequent packets. As an example, consider the parallel application of the Random Value and Add Redundancy patterns: A covert channel sender could modify the Identifier in the IPv4 header as well as it attaches an IPv4 option used to carry additional hidden data. A parallel application of a particular set of patterns to a single packet may not always be possible. In future work we will determine dependencies between patterns so that feasible pattern combinations can be identified. Much simpler is the sequential application of different patterns to subsequent packets. For instance, for one packet the Value Modulation pattern could be used to modify an unused field and LSB value modulation could be used for another field for the following packet.

Pattern Hopping

Sequential pattern combination uses a simple linear combination, which could be easily detected. To make detection more difficult, we propose a simple mechanism based on the concepts of protocol hopping covert channels (Wendzel and Keller, 2011) and synchronized random number generators similar to (Gianvecchio et al., 2008).

Let P be a set of patterns. The sender S and receiver R agree to use a certain cryptographically secure pseudo-random number generator (CSPRNG) with a certain seed value V. The agreement on CSPRNG and V has to be done

separately over a secure transmission channel, since the covert communication could be easily uncovered if both are known. The sender S and receiver R initialize their CSPRNG with V. Both CSPRNGs now are synchronized. Let t be the sequential number of the transferred pattern, incremented each time a packet is being sent (and received). t is initialized with 0. For example, t could also be the timestamp or the sequence number of a packet (immutable between S and R). S chooses $p_i \in P$ where i = CSPRNG(V, t) mod $|P|$ and applies p_i to the actual packet to send. R knows the pattern since R gets t and knows V and CSPRNG. Thus, the patterns used are randomized. Instead of using a pattern for each packet we can also increase the modulus so that 'unmapped' patterns are ignored (limiting the bandwidth).

As network packets, and with it, the transfer of covert data, can be error-prone due to packet loss or re-transmissions caused by transport layer protocols, a reliable communication is a necessity to prevent the de-synchronization of the CSPRNG. To overcome this synchronization problem, so-called *micro protocols* can be used, i.e. covert channel-internal control protocols with reliability features (Ray and Mishra, 2008). Micro protocols have been well studied over the last years and optimized micro protocols are available (Wendzel and Keller, 2012c; Backs et al., 2012).

Moreover, the selection of patterns can be done with adaptive techniques (Yarochkin et al., 2008). Adaptive covert channels dynamically customize the use of covert channel techniques in order to bypass blocked communications (Yarochkin et al., 2008) and thus provide a more reliable data transfer. In combination with micro protocols and pattern hopping, adaptive covert channels could not only switch between network protocols but between patterns in case a technique or pattern will be administratively blocked. While it is comparably easy to block a specific covert channel technique, it is more challenging to eliminate a covert channel that switches to a different *pattern* if one pattern was blocked, as this requires implementing countermeasures against multiple patterns.

4.6 Towards Pattern-based Countermeasures

Protection techniques for covert channels either aim on eliminating a covert channel, limiting a channel's capacity or detecting a covert channel (Zander et al., 2007a). The research community considers covert channel protection a *challenging task* (Gianvecchio and Wang, 2007) and due to the large number of existing covert channel techniques, it is currently difficult to counter *all* covert channels in practice.

Previous approaches only targeted selected covert channels in a given network protocol. The introduction of patterns which comprise a generic description of a hiding technique enables a more practical approach to develop countermeasures for a whole set of hiding techniques linked to the same pattern. As our 11 patterns represent at least the 109 discussed covert channel techniques, a comparably small number of approaches would be enough to counter these covert channel techniques. Thus, the integration

of pattern-based countermeasures will lead to a significant reduction of necessary protection mechanisms in practice.

On the other hand, some specific countermeasures may be more effective than a more general technique that can counter *all* covert channels associated with a particular pattern. For instance, some countermeasures are optimized for detecting hidden data in a header field of a particular protocol. The direct adaption of such a countermeasure to another network protocol, where the particular header field is linked to a different value distribution, may lead to a lower detection accuracy. Therefore, countermeasures for patterns require a *variation* (analog to Section 4.5) as well in order to adapt them to particular network protocols.

4.6.1 Countermeasures for Patterns

In order to evaluate the applicability of existing countermeasures to patterns, we focused on countermeasures covered by the surveys of Llamas *et al.* (Llamas et al., 2005) and Zander *et al.* (Zander et al., 2007a). Our evaluation does not include measures that focus on local covert channels (e.g. the *fuzzy time* approach (Hu, 1991)) or on the prevention of covert channels at the time of system development (e.g. the *shared resource matrix methodology* or *covert flow trees* (Kemmerer, 1983; Porras and Kemmerer, 1991)). In particular, we considered traffic normalization, the network pump, statistical approaches, and machine learning approaches.

Traffic Normalization (TN)

Traffic normalizers remove ambiguities and policy-breaking elements in network traffic, , which makes them effective especially against storage channel patterns. The application of normalizers results in side-effects as the normalization of PDU headers often includes setting header fields to default values, i.e. these fields are not usable anymore (Handley et al., 2001). Existing traffic normalizers are, for instance, the *network-aware active warden* (Lewandowski et al., 2007), the *Snort normalizer* (Snort Project, 2012), and *norm* (Handley et al., 2001) which, taken together, provide more than 100 normalization techniques. Literature divides normalizers into *stateless* and *stateful* normalizers. Stateless normalizers focus on one packet at a time, and they do not take previous packets into account, while stateful normalizers cache information of previously received packets to evaluate traffic in a more advanced manner and can also detect *more* covert channels, as shown in (Lucena et al., 2006).

For always legitimate values (e.g. allowed values for the IPv4 *protocol* field or allowed destination addresses in LAN frames), the creation of normalization rules is challenging and linked to constrictive side-effects (Fisk et al., 2003), which makes normalization ineffective against different forms of the *Value Modulation pattern*[14].

[14]On the other hand, normalizers can eliminate TTL-based covert channels as described in (Zander et al., 2006) by setting the TTL value of all packets to the same value.

A problem of traffic normalizers is their limited buffer size (Handley et al., 2001): Buffers cache packets of data flows to re-assemble these flows. Normalization techniques which require large buffers, for instance, to re-order network packets (*PDU Order pattern*) or to normalize the inter-arrival timings and data rate (*Inter-arrival Time pattern and Rate pattern*), are only useful as long as the normalizer's resources are not exhausted. Another problem, especially in IP networks, is that traffic can take different routes and thus, not all packets of a connection pass a normalizer which can result in incomplete information about connections (e.g. missed packets of TCP handshakes (Handley et al., 2001)).

Traffic normalization can only be applied to all network traffic ('blind' normalization) if the normalization is transparent (it does not affect the traffic significantly). Hence, blind normalization cannot eliminate all covert channels. However, if accurate detection methods exist, detected covert channels can be eliminated or limited using targeted normalization or even disruptive measures, e.g. the overt traffic could simply be blocked.

Network Pump and Related Concepts (NPRC)

Techniques to limit the capacity of network covert timing channels based on the *Inter-arrival Time* and *Rate patterns* (e.g. the pump (Kang and Moskowitz, 1993) and the ACK filter (Ogurtsov et al., 1996)) are traffic normalizers as well. These countermeasures either prevent the entire data flow from HIGH to LOW or do only allow the transmission of acknowledgement messages from HIGH to LOW (related to a data flow from LOW to HIGH).
Another approach presented by Wendzel and Keller (Wendzel and Keller, 2012b) limits the covert channel discussed in (Wendzel and Zander, 2012) (*Value Modulation pattern*) which encodes hidden information in the sequence of utilized network protocols. The approach is called a *protocol switching-aware active warden* (PCAW) and introduces delays on protocol switches.
As in the case of traffic normalization, buffer sizes and routing effects limit the capabilities of the NPRC approaches.

Statistical Approaches (SA)

The detection of covert timing channels based on inter-arrival times was achieved by Cabuk *et al.* (Cabuk et al., 2009) by representing recorded, rounded inter-arrival times as strings. The strings are compressed and the *compressibility* of a string is used as an indicator for the presence of a covert timing channel. The method takes advantage of the fact that covert timing channels generate traffic with a few characteristic inter-arrival times to signal the different covert bits and thus, result in a few similar strings which can be compressed more efficiently than normal random inter-arrival times.
Cabuk *et al.* presented two additional statistical approaches, one based on the calculation of the ϵ*-similarity* and another based on the standard deviation of recorded inter-arrival times (Cabuk et al., 2009).
Berk *et al.* developed another detection approach for inter-arrival time

channels (Berk et al., 2005). Their technique uses the fact that timing channels generate inter-arrival time distributions that differ from inter-arrival time distributions of normal application traffic. Gianvecchio and Wang showed that covert timing channels can be detected with high accuracy by analyzing the entropy of inter-arrival times (Gianvecchio and Wang, 2007).

While statistical approaches have been effective primarily against channels based on the *Inter-arrival Time pattern* and the *Random Value pattern*, their application to all other patterns is thinkable, since the use of all covert channels leads to changes of statistical value distributions.

Machine Learning (ML)

Covert channels can be detected using supervised ML approaches where some statistical features are used to characterize covert channels and normal traffic. Classifier models are then trained based on provided examples of features of covert channels and normal traffic. Sohn *et al.* demonstrated that simple covert channels encoded in the IP ID or TCP ISN field can be discovered with high accuracy by Support Vector Machines (SVMs) (Sohn et al., 2003). Tumoian *et al.* showed that a neural network can detect Rutkowska's TCP ISN covert channel (Rutkowska, 2004) with high accuracy (Tumoian and Anikeev, 2005) (both *Random Value pattern*). Zander *et al.* demonstrated that inter-packet timing channels can be detected by C4.5 decision trees trained on several features (Zander et al., 2011) (*Inter-arrival Time pattern*). Wendzel and Zander showed that C4.5 decision trees also detect simple protocol switching channels (*Value Modulation pattern*) with high accuracy (Wendzel and Zander, 2012). Besides the mentioned patterns, the application of ML to detect all other patterns appears possible.

Applicability of Countermeasures

Table 4.3 summarizes our findings on the applicability of the discussed countermeasures in the context of covert channel patterns. We did not only take existing applications of countermeasures but also potential applications into account, since no pattern-specific implementations are available yet. In general, the prevention of covert channels is always feasible if *all* traffic is blocked but such approaches are not applicable in practice or demand a high-quality covert channel detection. Therefore, Table 4.3 does not cover techniques which block the entire traffic.

4.6.2 Illustration: Traffic Normalization

Although traffic normalization was never discussed in the context of covert channel patterns, the existing normalizers already have pattern-oriented capabilities. As dozens of normalization techniques exist, we will discuss only selected ones to show that pattern-oriented countermeasures are feasible.

	Elimination	Limitation	Detection
Storage Channel Patterns			
P1. Size Modulation			SA/ML
P2. Sequence	TN		SA/ML
P2.a. Position	TN		SA/ML
P2.b. Number of Elements	TN		SA/ML
P3. Add Redundancy	TN		SA/ML
P4. PDU Corruption/Loss	TN		SA/ML
P5. Random Value	TN		SA/ML
P6. Value Modulation		TN (limited), NPRC	SA/ML
P6.a. Case	TN		SA/ML
P6.b. LSB	TN		SA/ML
P7. Reserved/Unused	TN		SA/ML
Timing Channel Patterns			
P8. Inter-arrival Time		TN (limited), NPRC	SA/ML
P9. Rate		TN (limited), NPRC	SA/ML
P10. PDU Order		TN (limited) NPRC	SA/ML
P11. Re-Transmission			SA/ML

TABLE 4.3: Application of Covert Channel Countermeasures to Patterns

For instance, existing normalizers can replace the TTL of IPv4 packets with a fixed value to eliminate covert channels. Traffic normalizers can apply the same technique to counter covert channels in the IPv6 *hop limit* field. The technique thus counters the *LSB pattern* even in case of a pattern variation to different network protocols. The LSB pattern could also be applied to the BACnet NPDU *hop count* field — no new normalization technique must be implemented for the same pattern.

Fisk *et al.* mention general cases for the application of traffic normalization (Fisk et al., 2003). For instance, unused fields can be cleared (*Reserved/Unused pattern*); decreasing fields (like the TTL) can be set to fixed values (*LSB pattern* and *Value Modulation pattern*); and derivate fields (i.e. those depending on other fields, like the *Checksum* or the *Internet Header Length* in IPv4) can be re-placed with correct values or the particular packets can be dropped (*PDU Corruption/Loss pattern*). Such general applications of normalization rules can be used in a protocol-independent manner and are thus capable of countering a whole set of hiding techniques which are based on the same pattern.

4.6.3 Illustration: Protocol Switching-aware Active Warden

The aforementioned PCAW (Wendzel and Keller, 2012b) introduces delays on protocol switches and thus limits the bitrate of covert channels that signal hidden information through the use of particular network protocols. As

shown in (Wendzel and Keller, 2012b), the PCAW cannot only be successfully applied to protocol switching covert channels based on IPv4 but also to building automation networks using BACnet and thus exemplifies the variation of countermeasures as well.

4.7 Conclusion

We evaluated 109 network covert channel techniques from the last decades and extracted abstract patterns from these techniques. We were able to represent all 109 techniques by 11 patterns which we arranged in a hierarchical catalog based on the *Pattern Language Markup Language* (PLML). Moreover, we showed that the majority of these techniques can be reduced to only four different patterns – evidence that many network covert channel techniques invented represent very similar techniques. If the scientific community uses hiding patterns (which include alias descriptions), these patterns can help to increase the scientific efficiency as re-inventions can be limited (Chapter 3.4).

The pattern catalog is provided on-line (*https://ih-patterns.blogspot.com*) in order to allow the scientific community to modify and extend the covert channel pattern collection in a moderated process. Our catalog eases the novelty evaluation of future covert channel techniques.

We presented the concept of *pattern variation* for covert channels. Pattern variation allows the automatic adaption of a generic hiding technique to different network protocols without requiring a re-implementation of the technique itself. Since covert channels are a dual-use good, we also introduced the pattern-based approaches pattern hopping and pattern combination to improve the throughput and stealthiness of covert channels.

If prevention approaches counter generic patterns instead of hiding techniques, the number of necessary countermeasures is greatly reduced. Under the assumption that future techniques for covert channels will often fall into one of the existing pattern categories, the value of our pattern-based approach increases even further. To this end, the implementation and evaluation of pattern-based countermeasures in practice is considered important future work.

Additional future work will comprise the generation of a PLML-based pattern catalog for *local* covert channels and for *payload-based* hiding techniques.

Part III

Pattern-based Research Methodology

Chapter 5

Pattern-based Novelty Evaluation

Abstract The last decades of research on network steganography led to more than hundred techniques for hiding data in network transmissions. However, the previous chapter has shown that most of these hiding techniques are either based on the same idea or introduce limited novelty, enabling the application of existing countermeasures to combat these techniques. In this chapter, we provide a link between the field of creativity research and network steganographic research. We propose a framework to help evaluating the creativity bound to a given hiding technique. This way, we support two sides of the scientific peer review process as both authors and reviewers can use our framework to analyze the novelty and applicability of hiding techniques. At the same time, we contribute to a uniform terminology in network steganography.

Originally published: Wendzel and Palmer (2015)

5.1 Introduction

In the previous chapter, it was shown that 109 network information hiding techniques developed between 1987 and 2013 can be reduced to only 11 different “hiding patterns", i.e., abstract descriptions of these hiding techniques. Moreover, it was shown that the majority (approx. 70%) of these 109 techniques is represented by only 4 hiding patterns. Although it must be noted that many of these techniques differ in slight detail, their novelty is limited. However, the amount of published techniques reflects a great demand of network steganographic techniques and stresses the need for continuously improved solutions.

A major drawback of having a high number of similar hiding techniques is that countermeasures can be adjusted easily to slightly new hiding techniques while it would be harder to create completely new countermeasures which have to deal with entirely novel hiding techniques. For this reason, the research community should foster such fully novel methods.

Moreover, while the number of publications presenting hiding techniques is steadily increasing, the chances for redundant techniques with similar basis increase as well — a problem which is also known by the patterns research community (Henninger and Corrêa, 2007). Another problem is that such redundancies lead to terminological inconsistencies as different terms can be used to describe similar (or equal) hiding techniques. A means to handle redundancies and terminological inconsistencies is to compile surveys on a regular basis.

Another problem of published hiding techniques are the huge differences in the explanation of novelty and usefulness. To provide an example, some researchers motivate the quality of their hiding techniques on the basis of the channel capacity while others highlight the fact that their technique is the first to hide data inside a new network protocol header. The divergence of such provided arguments allows no comparison and hinders experimental verification by other peers.

In this chapter, we provide a framework and a metric for evaluating the creativity in network steganography research. In line with creativity research, creativity comprises the novelty and applicability of products (solutions). Based on our creativity framework and metric, the contributions of this chapter are:

- *Long-term improvement of terminology*: Our framework deals with the problem of terminological inconsistencies by providing a step-by-step approach on the basis of the hiding patterns presented in (Wendzel et al., 2015). A key aspect in this regard is the handling of redundant hiding techniques.
- *Creativity evaluation for hiding techniques*: Our framework contributes to two aspects of the academic peer review process. Firstly, it helps authors to clearly underpin the creativity, i.e., novelty and applicability, of their proposed steganographic hiding techniques. Secondly, it helps reviewers to evaluate the creativity of steganographic hiding techniques. Therefore, the framework introduces a novelty metric.

- *Applicability in practice*: The proposed framework is designed to fit the needs of the actual workflows in academia and is thus embedded into the academic peer review process rather than being a theoretical discussion.
- *Support for novel hiding techniques*: Moreover, our framework fosters the creation of entirely novel, highly creative hiding techniques by giving the most creative researchers the chance to publish new patterns.

The remainder of this chapter is structured as follows. Section 5.2 discusses related work and draws the link between creativity research and network steganography; in addition, it provides a background on patterns and presents lessons of the pattern community to be taken into account for our work. Section 5.3 presents our pattern-based creativity framework while Section 5.4 introduces the metric to evaluate the creativity of a hiding technique. Section 5.5 provides an exemplary walk-through for the creativity framework. We provide a discussion of our framework in Section 5.6 and conclude in Section 5.7.

5.2 Background

We first discuss related work and afterwards describe the link we draw between creativity and network steganography followed by the link between patterns and network steganography. Thereafter, we describe the lessons learned from the pattern community.

5.2.1 Related Work

Terminology of computer security was discussed by a number of authors, (e.g. Brinkley and Schell, 1995; Freiling et al., 2014). The terminology of information hiding and its sub-discipline, network steganography, was discussed in additional publications (see Petitcolas et al., 1999; Zander et al., 2007b; Lubacz et al., 2014). Although a systematic categorization of hiding techniques is presented, these publications on network steganography lack the discussion of underlying patterns. The previous chapters provide the only known surveying content on network information hiding using patterns.

Our aim is not to provide a new approach for knowledge management, as a plethora of work on especially organizational knowledge management and frameworks is already available (Alavi and Leidner, 2001). Our work is also not the first that applies patterns for knowledge management. Henninger and Corrêa discussed advantages and drawbacks of a pattern-based knowledge management already in (Henninger and Corrêa, 2007) and we highlight their relevant outcomes in 5.2.4. Other approaches for scientific knowledge and terminology management are especially *i)* publication of ideas in textual form, without using the structure of patterns; these publications include books, journal articles, conference papers, technical reports, Wikis and other forms, and *ii)* publishing ideas in structured form, e.g. in

databases. These approaches are also useful for a framework for creativity evaluation but due to the availability of hiding patterns of (Wendzel et al., 2015) and (Mazurczyk et al., 2016a) and the high level of knowledge on patterns within the computer science community, we have chosen patterns as a basis for our framework.

However, our work is the first that is tailored to match the requirements of network steganography research by providing a metric to evaluate and compare the novelty and applicability of hiding techniques while also enabling a unified terminology and providing an integration into the peer review process.

5.2.2 Bridging Creativity and Network Steganography

Described briefly, creativity is 'adaptive originality', i.e., creativity requires the generation of an idea that is both original as well as adaptive to a particular context (Simonton, 2011; Amabile, 1983; Csikszentmihaly, 1996; Drazin et al., 1999; Farmer et al., 2003). Creativity is seen as one of society's biggest assets (Simonton, 2011). Overlapping the fields of science, technology, economics and arts, creative efforts lead to competitive advantages, innovative products and processes and are honored by awards like the Nobel prize (Agars et al., 2012; Caroff and Lubart, 2012).

Being such an important driver for social and economic progress and wealth, it is obvious that a great deal of research has focused on the 'creativity phenomenon' (Mumford and Gustafson, 1988; Runco, 2006; Ward, 2004). Different sciences are dealing with the topic, such as engineering, sociology, and psychology to only name the most relevant for our purpose. Due to varying research backgrounds and different research focuses and aims, conceptualizations of creativity (e.g. creativity as logic, genius, chance, and zeitgeist) differ and so do the findings. Smith counts more than 100 definitions of creativity (Smith, 2005), and his focus is limited to psychological literature only. Thereby it is not surprising, that Simonton emphasizes the inconsistencies of research on creativity (Simonton, 2004).

However, the diverse approaches to creativity can be categorized by their primary focus. In this tradition the '*4p*' classification is used: creativity as *product, person, process* or *press* (Mooney, 1963; Rhodes, 1961):

- *Product*: As mentioned above, for a product to be labeled creative, two characteristics are indispensable: it has to be original as well as adaptive. Original refers to: new, unusual, unique, new viewpoints, varied, breaking from existing patterns. Adaptive means: useful, valuable, effective, efficient, and contributing to society (Palmer et al., 2015).
- *Person*: In psychology a strong focus lies on the identification of individual traits fostering creativity. A combination of cognitive (intelligence, knowledge), non-cognitive (personality) and motivational (need for creativity, interests, achievement motivation) abilities and predispositions determines creative potential on an individual basis (Palmer, 2015).

- *Process*: Especially in highly experience- or knowledge-driven domains, creative ideas and innovative solutions do not emerge out of a sudden. Process research deals with the challenges one faces on his way from initial recognition of a problem or demand to a sustainable and accepted creative solution and describes the relevant traits and behavior in respective process phases. Psychological literature proposes several process models (for an overview see (Palmer, 2015; Howard et al., 2008)). An initial understanding of the creative process distinguishes four process stages: preparation, illumination, verification, and implementation (Lubart, 2001).
- *Press*: 'Press' comprises the creative environment. This research stream investigates the variety of situational and social contextual factors influencing the creative success, such as motivation, scope of action, leadership style and required support. For an overview of influencing variables in an organizational setting, see (Krause, 2013), (Oldham and Cummings, 1996), (Schuler and Görlich, 2007) or (Zhou and Shalley, 2003).

Network steganographic research describes, develops, and evaluates hiding techniques and thereby focuses on *products* according to the *4p taxonomy*. In the field of network steganography, many research is available, i.e., a high total output due to the number of publications is present. And, of course, the amount of publications as well as their quality (e.g. measured via citation index) can be used to quantify the impact of a person's productions (Simonton, 2011) — measures also applied to *rank* researchers. In this chapter, we primarily focus on the creative products rather than on the creative person, the creative process or the creative press.

As mentioned, a large extent of publications on hiding techniques provides a rather small novelty as they belong to the same hiding patterns. To introduce a bridge between creativity and network steganography, novelty and applicability as key characteristics of creative products must be taken into account. We present a metric to evaluate both aspects.

5.2.3 Bridging Patterns and Network Steganography

So far, patterns were only applied in network steganography as a means to create a taxonomy and survey of network hiding techniques (see previous chapter). In network steganography, the general problem is to hide information within the context of a network transmission; however, many solutions for this problem-context pair exist, which results in a number of patterns.

Patterns are linked to a several advantages, including their flexibility, the fact that they can serve as an easy basis for the expression and discussion of ideas, and their easy structure (May and Taylor, 2003). These advantages make patterns a known tool for knowledge management, also outside of science in business and governmental organizations (May and Taylor, 2003).

In general, patterns can be described in different forms, which complicate their comparison and understanding. For this reason, common *languages* were developed to describe patterns in a unified manner. A well-known and established pattern language is the *pattern language markup language* (PLML) (Fincher, 2003). All pattern languages comprise own attributes used to describe a pattern. In PLML, a pattern comprises a number of XML-based attributes, of which only a few are of significance for hiding methods (these were introduced in Table 4.1 in the previous chapter).

Using such pattern languages, patterns can be easily grouped in a *pattern collection*. A pattern collection comprises multiple patterns belonging to a similar domain, e.g. all patterns that describe hiding methods for networks.

As stated by Henninger *et al.*, one of the main intentions of patterns is to *provide a common vocabulary by which people can succinctly communicate well-known solutions to recurring problems* (Henninger and Corrêa, 2007). A so-called *pattern collection* is a *set of patterns addressing a fairly cohesive problem domain* (Henninger and Corrêa, 2007). Patterns for the problem domain of hiding techniques can thus be considered a pattern collection.

5.2.4 Learning from the Software Patterns Community

While pattern collections offer a clear advantage of cleaning up a domain and helping to share *ideas and abstractions between people working in the same conceptual space* (May and Taylor, 2003), they are also linked to a number of challenges (Henninger and Corrêa, 2007).

Firstly, pattern collections are stored in a multitude of locations, e.g. conference proceedings or on the web (in wikis and websites of different format). For this reason, patterns are not easily accessible as many potential sources must be used to find new patterns. Secondly, pattern collections are not easy to link as different styles for pattern collections exist, which describe patterns with a different set of attributes. Thirdly, duplicates of patterns can arise under different names. On the other hand, Henninger mentions the need for duplicates as *people may want to express the patterns different[ly]*, and patterns should allow *a certain degree of expression* (Henninger and Corrêa, 2007). Our creativity framework addresses all three issues as we will explain in the next section.

5.3 A Pattern-based Framework for Network Steganography

We first illustrate the requirements of our creativity framework and then present the framework itself by explaining all the steps that must be performed within it.

5.3.1 Requirements

Our aim is to learn from the known drawbacks of pattern collections which we discussed in the previous section and to adjust our framework to scientific practice. Hence, we compiled the requirements to address these drawbacks in our creativity framework:

1. The framework must require a researcher to publish his patterns in a publicly accessible way to ensure that all patterns and all updates to the pattern collection are accessible to the scientific community.
2. The framework must emphasize the need for a unified form of pattern description, which is the *pattern language markup language* (PLML).
3. The framework must address the aspect of duplicates; PLML contains an attribute that allows aliases for existing patterns.
4. To find use in practice, the framework must comply with the common workflow of scientific research, especially within a peer review.
5. The framework must not rely on the participation of all scientists of the research community. It is unlikely that a whole community will accept and work according to the same framework as people may not even aware of the framework or reject its idea.
6. The framework should foster the development of new and applicable, that is creative, hiding methods.

5.3.2 Creativity Framework

Our creativity framework is visualized in Figure 5.1 and consists of five steps, which we will explain one after another. Each step in the framework is performed by at least one role; either the scientific community (C), which performs peer reviews and maintains the pattern collection, or the researcher(s) (R), which publish novel network steganographic techniques. If a step of the framework is performed by both roles (C and R), the leading role is highlighted in Figure 5.1.

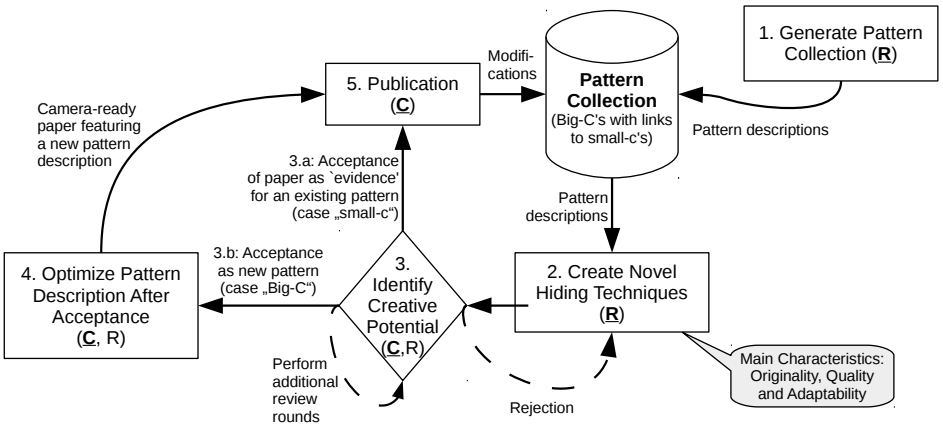

FIGURE 5.1: Creativity framework for network steganography

Step One: Generate Pattern Collection

Before a pattern collection can be used and serve as basis for further research, it must be created by one or many researchers (R). This step is performed by compiling a survey in which recurring design principles of hiding techniques are analyzed and grouped into hiding patterns. These patterns must be explained in the form of a pattern language – in our case PLML – to provide a unified and clear structure of the grouped hiding techniques. The compilation process for a new pattern collection may require several months of work. By publication in a public journal, the pattern collection becomes available to the scientific community (C).

The hiding patterns described in (Wendzel et al., 2015; Mazurczyk et al., 2016a) are suitable for the initial pattern collection. Due to the detailed description of the approach, these publications can also serve as a guideline for the development of new pattern collections.

Step Two: Create Novel Approaches

Completely new and thereby highly creative ideas are quit rare. In fact, a closer look at creative contributions shows that a lot of innovative products on the product market, theories in science or contributions in literature and arts are reinterpretations or advancements of already existing concepts. For example, Kepler proposed his three laws of planetary motion after he drew an analogy between planetary observations and magnetism (Gentner et al., 1997). Accordingly, creativity research focusing on the process perspective also emphasizes the importance of conclusions by analogy. Many models of the creative process include a distinct process stage often labeled *category combination*. Category combination comprises the combination or reorganization of knowledge (respectively, information) for the generation of new ideas (Baughman and Mumford, 1995; Mumford et al., 1997; Mumford et al., 2003; Mumford and Gustafson, 1988).

Various offensive as well as defensive cyber security techniques, including firewalls, intrusion detection systems, honeypots, botnets, and worms, follow ideas whose equivalents can be found in nature (Mazurczyk and Rzeszutko, 2015). Not only does their technical logic represent an adaption of proven evolutionary concepts, but also the naming of these techniques stresses their close link to already existing mechanisms.

Applying concept combination from creativity research to the area of network steganography, researchers (R) should thus keep the patterns of the pattern collection in mind when designing new hiding techniques. For instance, an existing pattern's technique could be inverted or combined with other patterns to build *ideas* for new patterns. Therefore, a researcher may select the most diverse patterns to produce higher novelty in comparison to extending ideas of hiding patterns with lower diversity. However, a sole parallel application of two patterns (e.g. a timing channel pattern combined with a storage channel pattern) cannot be considered a *new* pattern as each technique has to be handled by a separate pattern. It may, however,

be imaginable to introduce hybrid patterns which combine existing patterns in useful ways. Such hybrid patterns must be considered as second tier patterns and cannot possess the same status like non-hybrid pattern as firstly, basically all patterns can be combined to form hybrid patterns, and secondly, the idea of hybrid hiding methods was already published (by Mazurczyk and Szczypiorski, 2008).

Indeed, a researcher may apply his very own discovery methods and can create entirely novel approaches without taking existing patterns as a basis. An uncommon approach to discover novel patterns may even result in higher creativity (Simonton, 2004). In this case of applying own discovery methods, the researcher may only use the pattern collection to ensure that his new hiding technique is not already represented by an existing pattern. In other words, the framework does not limit the creativity and freedom of researchers but aims to support it.

However, all proposed hiding techniques of a researcher must fulfill the requirements of a creative product. Firstly, the new techniques must be original with respect to the existing patterns; secondly, they must be adaptive, i.e., applicable in practice, and must be proven by an implementation.

Step Three: Identify Creative Potential

In the form of a scientific paper that features a PLML-based description of the hiding technique, the researchers (R) submit their pattern (or evidence for an existing pattern) for peer review to the research community (C).

Both, C and R, evaluate the creativity of the proposed technique, while R initiates the process. Firstly, R evaluates his hiding technique (the creative product) based on his *own internal criteria for what can be considered a promising idea* (Simonton, 2004). If R concludes that his idea matches the necessary quality requirements for a scientific contribution, he needs to prepare a scientific manuscript for submission. In the manuscript, he underpins the creativity of his new technique using the creativity metric we introduce in Section 5.4. Afterwards, R submits the manuscript for peer review.

The peers (C) rate whether the stated arguments of R are actually true by performing a review, e.g. as described in (Smith, 1990). Within the review, the peers additionally evaluate the creativity of R's work by applying the same metric of Section 5.4.

This metric will help to distinguish between contributions of a high level of creativity and such contributions that improve our understanding of steganographic techniques but do not widen the existing pattern collection. The classification of new ideas, products or concepts by their level of creativity is quite popular in creativity research. In Psychology, the term 'small-c' is used to express that a creative product is linked to low creativity (Simonton, 2014; Silvia et al., 2014; Kaufman and Beghetto, 2009; Kaufman and Beghetto, 2013). In such a case, the work represents an *existing* pattern and can be added to the references of the particular pattern, i.e., R's proposed hiding technique is added to the pattern collection in order to provide *evidence* for an existing pattern. Therefore, R must (monitored by

C's peer review) explicitly reference the existing pattern, which leads to step five. Indeed, R is able to withdraw and re-submit his work to another conference or journal if he does not accept the small-c categorization.[1]

In case of a high level of creativity, the term 'Big-C' is used. In such a case, a new pattern can be created and added to the collection, which leads to step four. In both cases, 'small-c' and 'Big-C', the publication of a paper containing a pattern (or adding 'evidence' to an existing pattern) automatically integrates R's contribution to the pattern collection.

Note that C can also reject the work of R, which means that no alternation of the pattern collection is achieved. Criteria for rating the creativity level of R's hiding technique are presented in Section 5.4.

Step Four: Optimize Pattern Description After Acceptance

In step four, which only applies for 'Big-C' cases, R's idea was accepted as new pattern and will be published. To this end, R optimizes the pattern description within the submitted paper based on C's review. This step is a part of the process in which R creates the camera-ready version of his paper.

Step Five: Publication (Maintenance of the Pattern Collection)

Within the network steganography community, research groups – forming the people behind the role 'C' – are maintaining the pattern collection, basically due to peer review in step three. A project website for long-term maintenance of network steganography patterns was set up that allows participants to discuss and question existing patterns and their evidence: *ih-patterns.blogspot.com*. Every author showing an accepted paper that adds a pattern or evidence to an existing pattern can request that his publication will be added to the webpage. This allows an easy access for the research community to the pattern collection.

However, a scientific research field may not only be driven by the power of few established research groups. It is thus necessary that a pattern collection is not only accessible due to scientific publications, but also *forkable*. Forking a pattern collection is similar to forking an open source software project. In the case of an open source project fork, the code is copied; the existing contributor's names remain but the copy of the code is from that date on edited independently from the original and the names of the new contributors are added to the code. In other words, the pattern collection can be copied and extended by other individuals which can be part of C or stay even outside of the previous C. This allows changing the rules applied to evaluate the creativity of the hiding methods and also the rules of the framework. However, the reputation of a pattern collection is represented

[1] While most journals allow to have multiple peer review rounds per submission, conferences often directly accept or reject a submission. For this reason, conference reviewers (C) must explicitly state that a paper should be accepted under the condition that *i)* R references a particular pattern and that *ii)* R declares his contribution as 'evidence', what should be monitored by the conference chairs (also C) as the reviewers will have no further control over the process.

by its acceptance by leading scientists in a field and by the publication in high-quality journals and conference proceedings.

One aspect that can lead to a fork are so-called *multiples*: Simonton highlights multiples which appear on a regular basis in scientific history in form of parallel discoveries or rediscoveries (Simonton, 2004). As mentioned, such multiples – or *duplicates* – also appear in the area of patterns (Henninger and Corrêa, 2007) but Simonton adds the aspect of *grade* to it (Simonton, 2004). The grade is the *number of rival claimants to the discovery or invention*. For instance, if n authors propose the same hiding technique, the associated pattern multiple's grade is also n. The scientific community, which is in charge of the pattern maintenance, takes care of spotting multiples and their divergent naming.[2] Therefore, pattern duplicates can be added as *aliases* to an existing pattern (step three) and members of C can act as R to propose such changes (step two).

5.4 A Network Steganography Creativity Metric

One of the core goals of our framework is to provide a metric for the creativity of a hiding technique. Such a metric cannot be built on a single aspect as hiding techniques in network steganography are linked to a variety of attributes.

Therefore, we provide a threefold metric, which requires a textual description within R's manuscript for each of the following categories. Of these three categories, the primary category is considered the most important and the tertiary category the least important to evaluate the creativity of a hiding method:

1. Primary category (*originality of hiding technique*): The major aspect to evaluate the creativity of a hiding technique is the extent to which it differs from the existing hiding techniques represented in known hiding patterns. Due to the large divergence of hiding techniques, a textual representation is necessary to explain this part. For instance, using a reserved flag of a network protocol cannot be considered novel as patterns already describe this technique.

2. Secondary category (*steganographic quality*): A researcher (R) can take various steganographic attributes into account to support the quality of his hiding technique. The classical aspects to highlight in this regard are the detectability, robustness, and bandwidth of a hiding technique, but it is also possible to highlight its steganographic cost (Mazurczyk et al., 2016b). An optimally prepared manuscript highlights all four attributes.

3. Tertiary category (*adaptability* or *novelty of application area*): A steganographic hiding method may be applied to a new area (e.g. to smart vehicle networks) that was not subject to a network steganographic

[2] It is important to emphasize the fact that creativity research differs between the origination and acceptance of ideas (Simonton, 2004; Hammond et al., 2011; Lubart, 2001) The research community may forget non-accepted research work over the years, which can lead to a *rediscovery* of the same idea by another researcher.

research before. A new area of application can also be a particular network protocol. However, a new area of application does not necessarily increase the creativity of the hiding method itself since it represents only the adaptation of an exiting idea to another context. Instead, the adaptation to a new area supports the addition of R's hiding technique to the 'evidence' attribute of a given hiding pattern.

In other words, the novelty of the application area depends on the time of a technique's presentation. For instance, several years ago, steganographic methods in smart buildings might have been a novel application area but after first publications arose in this context, the novelty of the application area is now lower. Similarly, hiding information in the IP header was a newer area of application in the mid-1990's than it is today.

In step three of the framework, the researcher (R) provides arguments in his paper to support all three categories while the reviewers (C) review the correctness and reasonability of the provided arguments. On this basis, C can decide whether the approach is a new pattern, 'evidence' for an existing pattern, or not novel enough or of not acceptable quality to become a part of a pattern collection. If C decides to allow R a modified re-submission of his work, an additional review round can be performed — possibly multiple times.

5.5 Exemplary Walk-though

For a better illustration, we now provide an example on how to use the creativity framework. We assume that both, R and C are aware of our framework. If neither R nor C is aware of the framework, the framework would simply remain unused by these individuals, providing no update to the pattern collection at all. However, publications not based on patterns can be integrated into pattern collections a posteriori by creating surveys or by adding evidence to online pattern collections.

Step one of the framework (the creation of a pattern collection) must only be performed if no pattern collection is already available. For this reason, R can use the existing pattern collection as the basis for the following steps.

In step two of the framework, R combines the concepts of a reasonable number of patterns in the hope to create a new pattern — as mentioned, a typical process to generate a creative product. Alternatively, R can create ideas for new hiding techniques from scratch without taking existing patterns into account at this step.

Finally, R ends up with a considerably useful idea, namely to signal hidden information by the 'position' of a packet corruption within a network flow. For instance, if the third packet in a network flow is intentionally corrupted, it represents a hidden symbol A while a corruption of the sixth packet represents a hidden symbol B.

Being aware of the framework, R can recognize that his hiding technique is a combination of the two patterns *PDU Corruption/Loss* and *Position*. The

PDU Corruption/Loss pattern represents hiding techniques which signal hidden data via packet loss or via corrupted network packets (e.g. those having invalid checksums). The Position pattern transfers hidden information by modifying the order of header elements (e.g. the order of header lines of in a HTTP request).

Based on his idea of a hiding technique, R implements a prototype to verify the feasibility of the technique and to evaluate its technical details such as the channel capacity. R compiles all relevant information about his hiding technique in form of a scientific paper. When describing the hiding technique in his paper, R either explains to which patterns the technique provides new evidence or why the technique cannot be associated with an existing pattern and, for this reason, necessarily represents a new pattern. Finally, R submits his paper for peer review to a scientific conference.

During the review process, R's paper is sent to several peer reviewers (C). The peer reviewers state that the major aspect of the proposed hiding pattern is the position of a given PDU in a network flow (in R's case, the important PDU is the one that contains an error). Therefore, the reviewers consider the hiding technique as being 'small-c', i.e., it provides additional 'evidence' to the Position pattern, but cannot be seen as a new pattern itself (still step three).

Based on all received comments of the reviewers, R finalizes his paper and submits a camera-ready version for publication. After the camera-ready version of the paper is published and if patterns were taken into account, the 'evidence' is officially provided to the existing pattern through the accessible paper (step five).

If R's proposed hiding technique would have been accepted as a new hiding pattern (Big-C) by the peer reviewers, the published paper would represent the description of a new pattern, eventually featuring an optimized description to match the reviewer's requirements (step four).

R can support the visibility of his hiding technique (being it a patern or an evidence entry to an existing pattern) in form of a comment on the existing pattern listings (e.g. *ih-patterns.blogspot.com*) or at any other place where a pattern collection is published to increase the distribution of research findings.

5.6 Discussion

To achieve our goal of a practical framework, we do not provide a low-level discussion on scientific creativity as can be found in (Simonton, 2004). For instance, our framework does not focus on the slight, and in some aspects unclear, differences between multiples and their opposite (singletons). If the scientific community decides to accept the proposed framework and makes the decision to change the aspects of it or to add the handling of more details of scientific creativity, the model is open for change.

We will first discuss whether our framework achieves the previously introduced framework requirements, followed by a description on how to

match the general requirement that a pattern cannot be called a *pattern* until at least three evidence cases are provided. We end this section with a note on alternative application areas for the framework.

5.6.1 Discussion of the framework's requirements

We address the requirement of publicly accessible patterns (requirement one) by requiring R to publish novel patterns in publicly accessible, peer reviewed organs and by providing the option to provide comments to the existing pattern collection on-line or on alternative websites every researcher can create himself.

The requirement of a unified pattern description (requirement two) is addressed by the use of PLML as it ensures unified descriptions by its predefined set of attributes (e.g. evidence, alias or solution).

Our framework limits duplicates (requirement three) due to the framework's integration into the academic peer review process in which duplicates may be spotted. In addition, requirement three is addressed by the use of PLML aliases. Aliases allow the a posteriori merging of redundant ideas published using different names, i.e., they can be applied if duplicates were not identified during the peer review.

The integration into the peer review process matches the requirement of the framework's applicability to scientific practice (requirement four). Given the framework awareness of R or C, the consideration of patterns for a new hiding technique can be enforced. At the same time, the traditional review procedures are kept. The overhead of checking whether a proposed hiding technique matches an exiting pattern is minimal due to the small number of patterns and their hierarchical order.

The framework is applicable even under the circumstance that only a minority of researchers of the network steganography domain will apply it (requirement five). Even if only applied by few researchers, their combined effort for a unified pattern collection will lead to a more unified terminology and less re-inventions for hiding techniques. However, the more researchers apply in the framework, the more efficient it will be. In that sense, the framework represents a living process and the number of researchers participating in it can change over time.

Our framework fosters the creation of novel hiding techniques (requirement six) by giving researchers the chance to contribute an own pattern to the pattern collection. On the other hand, our concept can lead to the situation that a proposed hiding technique becomes a pattern although it should only be considered as 'evidence' for an existing pattern. For instance, one researcher could propose a new pattern and an unqualified reviewer might accept it as such although it should only be accepted as an 'evidence' entry for an existing pattern (or not at all) — this contradicts requirement three and is an existing problem of the network steganographic field that the framework will not solve but reduce if applied correctly.

Creativity research focusing on creative products outlines the problems related to creativity ratings of products like hiding techniques. Whenever

products are rated, this evaluation underlies a social construction of criteria. Which solution is considered as non-, lowly or highly creative depends strongly on the raters' background. Their individual expertise, experience and strategy influence ratings as well as the zeitgeist, cultural factors or simply a changing technological context in change of time. Little expertise or experience of the raters is one of the key issues. Rather uninformed reviewers might not be aware of already existing techniques. In consequence, they might not be able to detect the linkage of an alleged new pattern to an already known pattern of hiding techniques.

To deal with such problems and the resulting inconsistencies, we recommend compiling pattern surveys every 3-10 years and publishing these surveys in recognized journals or conferences. New surveys can modify the pattern collection by publishing a new version of it. Such surveys can, for instance, move a 'pattern' into the 'evidence' attribute of an existing pattern. The proposal of regular survey compilation matches the requirements four and five for integration into scientific practice as it keeps the pattern collection updated and provides good knowledge management for the scientific community as well as for practitioners. In addition, a scientific discussion about patterns should be performed using moderated websites as these can be updated at any time while surveys remain as mid-term and long-term solutions.

5.6.2 Pattern's Requirement of Recurring Designs

Patterns must – per definition – occur multiple times. A typical boundary value for a design to become a pattern is three occurrences, provided as references in the 'evidence' attribute of PLML. It is our belief that the scientific community should ensure that all hiding techniques with 'Big-C' are represented by new patterns and rediscoveries are kept at a minimum. For this reason, we suggest that for the few new patterns to be found, patterns with a 'pending' status are created. Such patterns are part of a pattern collection as all other patterns but feature the keyword 'pending' in their name. A simple heuristic could be to consider the 'pending' keyword obsolete as soon as three references are listed in the 'evidence' attribute. An actual removal of the keyword cannot be done a posteriori in the same publication but by the above-mentioned regular surveys.

5.6.3 Applicability in Other Areas

We assume that our creativity framework can also be applied to other areas of information hiding besides network steganography. An imaginable area is digital media steganography. Moreover, our framework can be used to create a pattern collection of steganographic countermeasures, which were also already linked to patterns in (Wendzel et al., 2015). For instance, various techniques similar to the *pump* (Kang and Moskowitz, 1993; Ogurtsov et al., 1996) can form a pattern while *traffic normalization* (Handley et al., 2001) must be considered a clearly different pattern due to its fundamentally different functioning. The initial step of creating a pattern collection

must be performed first in these areas. Finally, the framework will be applicable to non-information hiding, even non-computer science, areas due to its generic approach.

5.7 Conclusion

We present a framework and a metric for evaluating the creativity linked to hiding techniques and to handle inconsistencies in the terminology of network steganography. The framework can be applied in practice and is thus embedded into the academic peer review process. By providing an exemplary walk-through, we have shown the applicability of the approach. The framework's design is not static and can be modified by the scientific community to fit their needs and it can be adapted to future developments. As patterns serve as basic elements of our framework, the framework takes advantage of their flexibility, accessibility and structure. In addition, the community benefits from the framework's application even if only applied by a minority of researchers.

A drawback of our framework lies in the fact that different researchers, especially early-stage researchers, may unknowingly allow the integration of a pattern that is only a representation of another hiding technique. We foresee regular surveys (every 3-10 years) as a clean-up solution for this drawback.

In addition to steganographic hiding techniques, the creativity framework is also applicable to other areas of information hiding, such as countermeasures or digital media steganography, as well as to other sciences.

Chapter 6

A Unified Description Method for Hiding Techniques

Abstract Based on the framework provided in the previous chapter and the concept of hiding patterns, hiding methods can now be categorized in a unified taxonomy and their novelty and applicability can be evaluated during peer review. However, mentioning patterns and evaluating the novelty of proposed hiding methods is not enough to advance the methodology in network steganography.

Until now, papers presenting new hiding methods describe these methods in arbitrary ways, making them difficult to compare. For instance, some publications describe classical channel characteristics, such as robustness and bandwidth, while others describe the embedding of hidden information. We introduce the first unified description of hiding methods in network steganography. Our description method is based on a comprehensive analysis of the existing publications in the domain. When our description method is applied by the research community, future publications will be easier to categorize, compare, and extend. Our method can also serve as a basis to evaluate the novelty of hiding methods proposed in the future.

Originally published: Wendzel, Mazurczyk, and Zander (2016)

6.1 Introduction

On the basis of hiding patterns, the previous chapter defined a new academic workflow for the creativity evaluation of network steganography methods. The key concept of this workflow is that if a new hiding method cannot be represented by an existing pattern, it comprises higher novelty than a hiding method that was already described by a pattern and is thus not novel. The evaluation process of hiding methods in conjunction with hiding patterns can be integrated into the traditional peer-review process but requires an author to explain why a hiding method is (or is not) represented by an existing hiding pattern. The approach of the previous chapter fosters the reduction of terminology inconsistencies as new publications will be aligned to existing pattern terminology and the improved peer review process eases spotting any inconsistencies.

In this chapter, we provide a twofold contribution: we performed an in-depth analysis of hiding method descriptions in papers and discovered several inconsistencies in these descriptions. We also provide a way to describe hiding methods that helps to prevent such inconsistencies by ensuring a unified, comparable description of the hiding methods.

Contribution A: Literature Analysis: We analyzed 131 hiding method descriptions from 74 publications published between 1987 and 2015. The analysis brought to light relevant inconsistencies in the *description* of hiding methods. In particular, we noticed large differences in the evaluations and technical descriptions between the publications. While for some hiding methods, the channel capacity is described, other publications focus solely on the embedding process, the application scenario of a hiding method or other aspects. In addition, the way in which hiding methods are described changed over time. Moreover, the descriptions of hiding methods even vary *within* some of the papers. We also found that some papers *combine* the evaluation and description for several hiding methods. For example, some publications discuss the overall throughput of multiple hiding methods instead of discussing the different channels separately. These non-unified descriptions make it difficult to compare publications that propose new hiding methods and to evaluate each described hiding method separately.

Contribution B: Presentation of a Unified Description Method: We introduce a method for a unified description of hiding methods in network steganography. Figure 6.1 visualizes our contribution in the context of previous work. The existing hiding patterns describe how hidden information is signaled, and the creativity framework provides a way to reduce redundant research outcomes and to evaluate whether a proposed hiding method is actually new, or not.

This chapter fills a missing gap in the previous work by introducing a multi-facetted and detail-rich description for hiding methods. In comparison to existing hiding method descriptions that we analyzed in *Contribution A*, our unified description has several advantages. When applied by the scientific community, our unified description method will enable the easy comparison, categorization, and evaluation of hiding methods. In addition, a unified description also eases the identification of research gaps as these can be

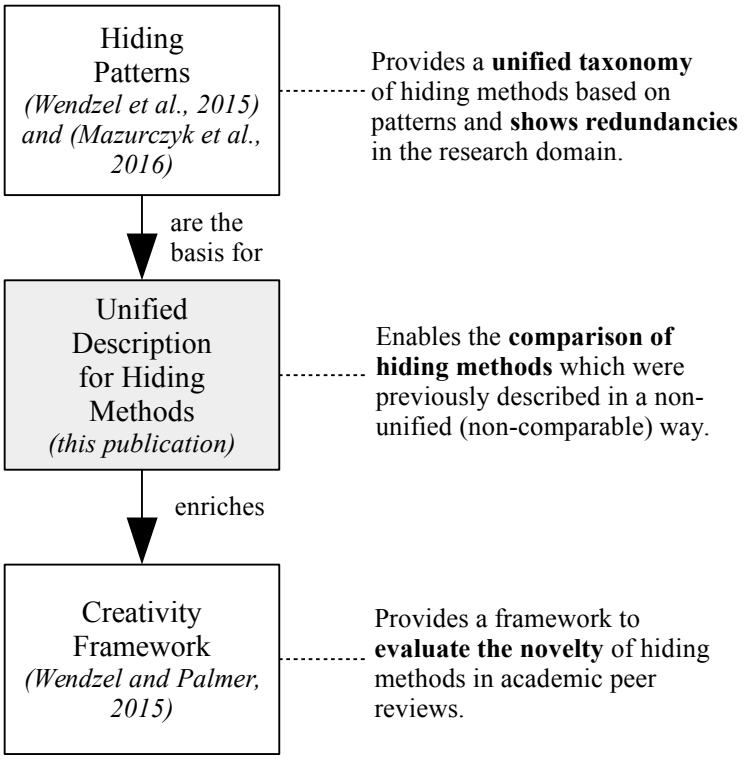

FIGURE 6.1: Contribution of this work: While hiding patterns serve as a basis for this new publication, it provides its own contribution by enabling the structured comparison of research work on hiding methods and it can be also used in conjunction with the creativity framework.

easily spotted (for instance, with the unified description method, a reader can easily see whether a channel capacity estimation is still lacking for a hiding method or whether a detection method is not given and must be subject to research first). Last, but not least, our description method can also serve as a basis for the creativity framework and will enrich the evaluation process of the same.

The remainder of this chapter is structured as follows. Section 6.2 provides an overview of the unified description method for hiding methods. The description method is split into three main categories, which are covered in Sections 6.3–6.5. We discuss results of our literature analysis in Section 6.6 and provide two exemplary descriptions in Section 6.7. Section 6.8 explains how the unified description can be used in combination with a creativity framework to evaluate the novelty of newly proposed hiding methods. We provide a conclusion in Section 6.9.

6.2 Unified Description Method

We analyzed the literature describing 131 hiding methods published since 1987. The analysis revealed that the descriptions of hiding methods in the related publications differ significantly regarding their provided information. To improve this situation, we designed a unified description method.

The unified description method can be directly applied to structure new scientific papers. This way, authors which present hiding methods make it easy for other researchers and reviewers to compare the new hiding method with existing hiding methods. Our unified description can be also combined with the 'creativity framework' (Wendzel and Palmer, 2015) to evaluate the *novelty* of a proposed hiding method by applying concepts of creativity research during the academic peer review process. By combining the creativity metric with our unified description method, both, the presentation of a hiding method and the underlying research novelty can be compared in a process that is unified, reasonable and re-constructable.

Our description of hiding methods is split into three categories, namely *general information about the hiding method*, *description of the hiding process*, and *potential or tested countermeasures*. The first two categories comprise subcategories and each (sub-)category can be mandatory or optional. Figure 6.2 provides an overview.

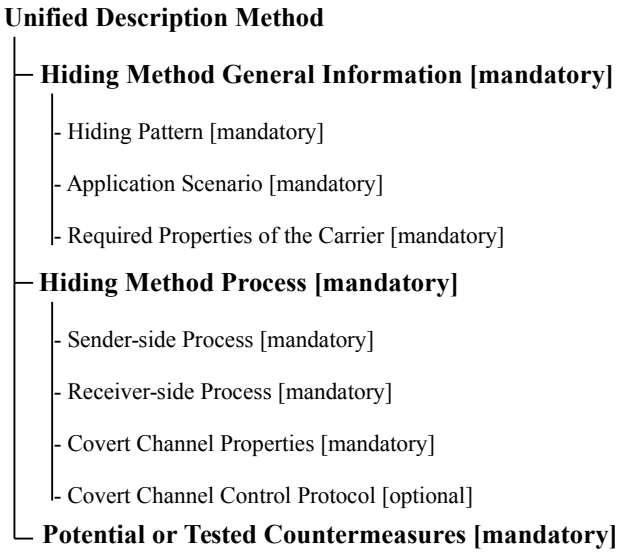

FIGURE 6.2: Overview of the description method's structure.

The category 'hiding method general information' consists of a link to an existing hiding pattern and a detailed description of the hiding method. It also includes a discussion of the application scenario and requirements of the carrier. The category 'hiding method process' is split into four parts: the sender-side and the receiver-side description of the hiding method, the details of the covert communication channel, and the description of an associated covert channel control protocol (if applicable). The third category discusses both potential and evaluated countermeasure, including those that detect, limit or prevent the particular hiding method's use. The following three sections explain all categories.

6.3 Hiding Method General Information

This section describes the general attributes of the steganographic method. These attributes include the hiding pattern that the method belongs to, the

considered and potential application scenarios, and general requirements for the carrier.

6.3.1 Hiding Pattern [mandatory]

In (Wendzel and Palmer, 2015), we proposed that when a new hiding method is to be published, it should be assigned to a particular pattern or provide evidence why it does not match any of the existing patterns. In case the proposed hiding method does not match any existing pattern, a new pattern can be created (Wendzel and Palmer, 2015). As the existing patterns are based on many hiding methods invented since 1987, it is likely that a new hiding method can be represented by an existing pattern. In the less likely case that no existing pattern fits, the authors of the new method must provide a detailed explanation of a new pattern that underpins the novelty of the hiding method they propose. The consequence of a new pattern is an extension of the existing pattern catalog.

If the authors decide a hiding method can be represented by an existing pattern, the pattern should be stated including the whole hierarchical path of the pattern in the pattern hierarchy (including sub-patterns). The hiding pattern hierarchy is described in detail in (Wendzel et al., 2015) and an online version is available under *https://ih-patterns.blogspot.com*.

A pattern name including the path within the hierarchy is the complete path from the root node of the hierarchy to the leaf that represents the pattern. For instance, a hiding method that modifies the least significant bit (LSB) of a header element would be represented by the "LSB" pattern and the full path of the hierarchy would be:

```
Network Covert Storage Channels
`-- Modification of Non-Payload
    `-- Structure Preserving
        `-- Modification of an Attribute
            `-- Value Modulation
                `-- Least Significant Bit (LSB)
```

For each element of the hierarchy, it should be explained briefly why the described hiding method belongs to the element, e.g. why the new method is a covert storage channel (and not a timing channel), why it preserves instead of modifies the structure of a PDU, why it is an attribute modification, value modulation, and LSB-based method. Using the hierarchy-based explanation, every reader who has knowledge of the pattern hierarchy can easily follow and verify the argumentation of an author.

6.3.2 Application Scenario(s) [mandatory]

This category describes the application scenario for which a hiding method was developed. It helps to identify novel application scenarios and makes it easier to compare different methods as some hiding methods may have application-specific limitations. Such methods may not be usable in other

scenarios. For example, hiding methods developed for breaking anonymization (Zander and Murdoch, 2008) may provide only small throughput making them unusable for general-purpose communication.

Many hiding methods were developed for general-purpose communication, i.e. the passing of secret messages between two or more parties. Typically this application is motivated by the existence of an adversarial relationship between different groups, such as government agencies versus criminal or terrorist organizations or dissenting citizens versus their governments. Other existing hiding methods are tailored to the case of hackers or corporate spies whose aim is to covertly control compromised systems or exfiltrate data from compromised systems. Similarly, malware, such as computer viruses or worms, can use hiding methods to spread undetected, to exfiltrate data, or for covertly exchanging information (e.g. execute brute-force attacks on cryptosystems (White, 1989)). Indeed, this rising trend has been confirmed by many real-life examples of information hiding-capable malware (Cabaj et al., 2018a; Mazurczyk and Wendzel, 2018).

On the other hand, there are hiding methods that were developed for very specific contexts. Some hiding methods were developed to break anonymization, for example Murdoch *et al.* developed methods to reveal servers hidden inside anonymization networks (Murdoch and Danezis, 2005; Zander and Murdoch, 2008). Other hiding methods were developed for transmitting authentication data, for example to allow authorized users to access open firewall ports while presenting these ports as closed to all other users ("port knocking") (deGraaf et al., 2005). Another type of hiding methods were designed for packet/flow traceback or watermarking – techniques used for linking different observable instances of network packets or flows in scenarios where packet contents cannot be used for linking (Houmansadr et al., 2009). Another specific application are hiding methods developed for cheating in on-line games (Murdoch and Zielinski, 2004).

In case a hiding method is for general-purpose communication, no comprehensive description is needed. However, for methods that were developed for a specific application in a specific context, the application scenario should be described in detail. Also, new application scenarios should be described in more detail than well-known scenarios.

6.3.3 Required Properties of the Carrier [mandatory]

This category is used to specify the properties of the carrier that the hiding method requires. It should describe whether the hiding method is limited to a certain protocol (or a service) as carrier or whether it can be used with several or even many different carrier protocols/services.

If the hiding method is tied to a single carrier protocol, the description must specify the protocol and describe the specific protocol features that are used by the hiding method. If a hiding method works with a set of carrier protocols, the description must specify the protocols and the protocol features the hiding method relies on. If a hiding method depends on certain protocol features that are common to a large number of protocols, the description must list the features and describe them. For truly generic hiding methods

that work with all kinds of carrier traffic the description may be short; however, in our experience such generic hiding methods are rare.

For hiding methods that are not only tied to certain protocols or protocol features, but also require certain operational conditions, these conditions must also be described. For example, a method that hides information by intentionally introducing packet losses assumes that packets of the carrier can be discarded. It can only blend in with the normal traffic if there is natural packet loss (Kraetzer et al., 2006); hence, the possibility and occurrence of natural packet loss is an operational condition for this hiding method.

6.4 Hiding Method Process

This section covers the categories which describe the actual process of the hiding method, including the embedding and the extraction of hidden data, as well as the channel properties and a potentially present control protocol.

6.4.1 Sender-side Process [mandatory]

This category describes the embedding process performed by the covert sender to hide secret data. It must be explained whether the sender is a centralized host/process or distributed. In the classical scenario, one sender transfers secret data to one receiver (the sender-to-receiver relationship is '1:1'). However, other scenarios are also possible and they depend on the specific context in which a covert transmission is performed or it can be a characteristic feature of the carrier utilized for hidden data exchange. In case of covert channel overlay networks, it is imaginable that one sender broadcasts the covert data to multiple receivers ('1:m'). In case of a distributed sending system, there may be n hosts forming one logical sender that transfers data to one or multiple receivers ('n:1' and 'n:m' relations). For instance, if the source address of a receiver indicates a hidden bit, two senders can be used to transfer a message of zero and one bits to a receiver.

The location of the covert data can be also centralized or distributed depending on whether the hidden data is 'inserted' into a single carrier (or a subcarrier) or it is distributed across several carriers (subcarriers). This means that the covert data can be embedded into one particular part of a packet or into multiple areas of a packet, but it can be also distributed among different flows (Mazurczyk et al., 2016a).

This category should also contain information that describe how the sender synchronizes *where* and *when* secret data is encoded. It is worth mentioning that synchronization capabilities can be necessary at two levels of a covert communication, the bit level and the packet/frame level. In case when multiple carriers or subcarriers can be selected for embedding secret data, a synchronization mechanism should be described, at least if it is necessary for the well-functioning of the covert channel. Such a mechanism should describe how the sender selects the carrier or subcarrier in a way recognizable by the receiver. If a timing channel is given, the category should accordingly explain how a synchronization of timing events is done.

It must be also specified whether the steganographic method generates its own cover traffic or whether data is hidden in third-party cover traffic. In case the hiding method is responsible for the cover traffic generation then a description of this process must be included here.

6.4.2 Receiver-side Process [mandatory]

This category describes the recognition and extraction process of the covert data at the receiver-side. Similar to the sender-side process description, the secret receiver can be also centralized or distributed and the same considerations apply here (see Section 6.4.1). If the receiver is a distributed system, it should be explained how the covert data is extracted from the hidden data carrier and how it is finally merged. If a synchronization mechanism was implemented by the sender, this category should describe how the receiver-side synchronization is performed.

6.4.3 Covert Channel Properties [mandatory]

In this category the considered hidden communication scenario(s) for the particular steganographic method should be described and it should be indicated whether the created covert channel is direct or indirect. Moreover, four characteristic features of the information hiding technique should be analyzed. This will allow to describe properties of the created covert channel.

For network steganography, two main possible communication scenarios may be considered, as illustrated in Figure 6.3. The first scenario, i.e. end-to-end scenario, is the most common: the secret sender and the secret receiver perform overt communication while simultaneously exchanging covert data. In this case the overt communication path is equal to the covert path. In the second scenario, i.e. Man-in-the-Middle (MitM) scenario, only a part of the end-to-end overt communication path is used for the hidden communication, as a result of actions undertaken by intermediate covert nodes. Therefore, the overt sender and overt receiver are, in principle, unaware of the steganographic data exchange. Obviously hybrid scenarios are also possible where the overt sender/receiver serves as secret sender/receiver but the other covert party is located in some intermediate node.

Next, it should be indicated whether the covert channel is *direct* or *indirect*, i.e. whether the overt traffic flows directly from the secret sender to the secret receiver or via one or multiple intermediaries. In case of a direct channel the overt traffic that contains the covert data flows directly from the secret sender to the secret receiver (who both can act as middlemen). A covert channel is indirect when the secret sender does not send covert data directly to the secret receiver (or a destination downstream of the secret receiver). Instead, the secret sender transmits the covert data to an intermediate host which then unknowingly forwards (due to the functions of the overt traffic protocol) the covert data to the secret receiver. This means that there are two flows of the overt traffic conveying the covert data, i.e. the first flow is between the secret sender and an unwitting intermediary and

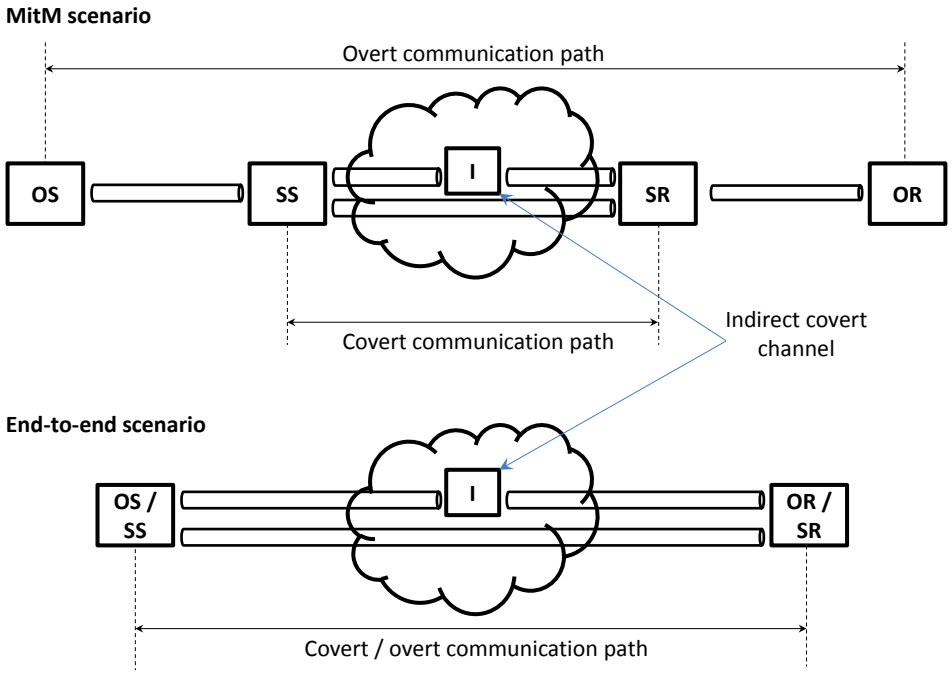

FIGURE 6.3: Hidden communication scenarios (OS – overt sender, OR – overt receiver, SS – secret sender, SR – secret receiver, I - intermediate node)

the second flow is between the intermediary and the secret receiver. Indirect covert channels provide an increased stealthiness as a warden does not observe a direct flow of information between the covert sender and receiver. On the other hand they are typically harder to implement and have a smaller capacity than direct channels. In case of an indirect channel the requirements on the intermediate need to be described.

For example, Rowland, 1997 proposed an indirect channel that exploits the TCP three-way-handshake. Instead of sending a TCP SYN segment with an ISN containing covert data directly, the secret sender sends the TCP SYN segment to the intermediary (a bounce host) with a spoofed IP source address set to the intended destination. The intermediary then sends a SYN/ACK or SYN/RST to the secret receiver with the acknowledged sequence number equal to the ISN+1. The secret receiver decodes the hidden information from the ACK number (ACK–1).

The classic characteristics of any information hiding method as mentioned by Fridrich in (Fridrich, 2009) are: *steganographic bandwidth* (or *channel capacity*), *undetectability* and *robustness*.

Steganographic bandwidth/capacity is the number of secret bits that can be sent per time unit when using a particular method or under a given condition, e.g. being undetectable with a specific detection technique, that is $B_s = b_s/t$. While the absolute bandwidth is most important, another relevant metric is the average ratio of the number of secret bits to the number of carrier bits $r_s = \sum(b_s)/\sum(b_c)$.

Undetectability is the inability to notice a steganogram within a certain carrier, under certain conditions, or the inability to notice the presence of a

steganographic communication, at all. A more formal description can be based on (Liu et al., 2010). Assume there is a sequence of N feature values used to encode covert data, such as N message delays or lengths, and define F and $\tilde{F}$ to be feature distributions for the covert channel and legitimate traffic respectively. Then, a covert timing channel is called polynomially undetectable with respect to a security parameter σ if it holds for any distinguisher D with a runtime polynomial in σ and for any N that is polynomial in σ that $D(F, \tilde{F})$ is negligible in σ.

Robustness is the amount of network and adversary noise a covert channel can withstand so that the covert receiver can still decode the data.[1] What constitutes noise depends on the type of channel, e.g. it could be delay, packet loss or packet modifications. Robustness can be measured as the capability to achieve a decoding bit error rate (BER) smaller than a given robustness threshold ϵ. Since the BER is inversely proportional to the Signal-to-Noise Ratio (SNR), robustness can also be defined in terms of the SNR. A channel is called γ-gain robust if the SNR after performing the encoding and modulation is γ times greater than the original SNR (Liu et al., 2010).

As proposed in (Mazurczyk et al., 2016b), another attribute is the steganographic cost, which describes the degradation or distortion of the carrier caused by the application of the steganographic method.

All these attributes should be described, if feasible. Especially for novel methods and for the description of third party tools, a comprehensive description of all four characteristics is hardly feasible. The author of a paper may have no access to implementations of these countermeasures, to testbeds in which countermeasures could be evaluated, or may have no knowledge of all existing countermeasures. Since the detectability issues are also discussed in the category 'Potential and Tested Countermeasures' only a brief description or reference to that category is required here.

6.4.4 Covert Channel Control Protocol [optional]

Several hiding methods utilize so-called covert channel control protocols. Covert channel control protocols embed small protocol headers providing several features including reliable data transfer (by introducing sequence numbers and ACKs), peer discovery and dynamic overlay routing (between steganographic peers), session management for covert transactions, adaptiveness and several features of application layer protocols (e.g. file transfer features) (Wendzel and Keller, 2014; Mazurczyk et al., 2016a). If utilized by a hiding method, both the design, implementation and features of a covert channel control protocol must be described in this category. Otherwise, the category can be left empty.

[1] Some previous work further distinguishes between *robustness*, robustness against normal network noise, and *active robustness*, robustness against noise created by an adversary, cf. Mazurczyk et al., 2016a.

6.5 Potential or Tested Countermeasures

This category comprises no sub-categories. It should describe potential as well as tested countermeasures available against the hiding technique. There are three types of countermeasures that can be applied against the covert channel created by a hiding method: elimination, limitation, and detection (Zander et al., 2007a). Not all of these three types may be applicable for a particular hiding method, for example some covert channels cannot be eliminated. The description must contain a discussion which countermeasures are applicable and which are not applicable including a justification. For instance, a reasonable justification for applying a countermeasure would be that it does not significantly influence the legitimate communication in a network. A more formal approach to explain the justification of applying a countermeasure would be to discuss its *Minimal Requisite Fidelity* (MRF), which is a *measure of the degree of signal fidelity that is both acceptable to end users and destructive to covert communications* and that was proposed by Fisk et al., 2003. Applicable countermeasures should then be described in detail.

Elimination means the covert channel created by a hiding method can be eliminated completely. For example, a method that hides data in unused header fields or padding bytes can be eliminated completely by a traffic normalizer that sets the unused header fields or padding bytes to a default value (e.g. zero). Some hiding methods cannot be eliminated, for example covert channels in on-line game protocols (Zander et al., 2008). If a covert channel can be eliminated, the description should include a discussion on how the elimination works and any possible limitations. The description should also include side-effects of the elimination process. For example, if a covert channel in a header extension can be eliminated by removing the header extension from the packets, then the description should include a discussion of the impact on the protocol functionality.

Limitation means the covert channels can be perturbed, for example by introducing noise, so that its capacity is greatly reduced and the channel effectively becomes useless. Limitation usually has side-effects on the carrier, so there is a trade-off between reducing a covert channel's capacity to a small value and not significantly impacting the carrier protocol. The description should include a discussion on whether a channel's capacity can be limited and the impact on the carrier should be characterized. If a channel can be eliminated then a description of a limitation method is optional.

Elimination or capacity-limitation are active countermeasures that require a warden to manipulate the network traffic (active warden) (Fisk et al., 2003). However, having an active warden may not be possible in every scenario.

The warden can audit the use of any covert channels it can detect. Usually detection mechanisms are based on some characteristics that can be observed, and the characteristics for traffic with covert channels are different from the characteristics of regular traffic (traffic without covert channels). While detection itself is passive, it can be coupled with active measures such as targeted blocking, elimination or limitation where the warden can manipulate suspicious traffic with more impunity. The description

should include whether the channel can be detected and outline the detection method. If the channel is impossible to detect, the description should provide a justification why it is undetectable. If a detection method is introduced, the proposed characteristics for identifying the covert channel must be defined.

If an evaluation of the proposed elimination, limitation or detection method(s) has been conducted, the description should summarize the evaluation scenario(s) and results. Ideally, an evaluation is done under realistic conditions, e.g. in real networks using realistic traffic, but in practice this is not always feasible. The description of the evaluation should point out any such limitations.

Another type of countermeasure is to change the specification of a network protocol to prevent its use as carrier in the future. For example, a network protocol prone to covert channels could be revised and updated with a newer version less prone to covert channels. In many cases this may not be realistic, as widely deployed protocols cannot be changed easily. However, in cases where an updated protocol could be realistically deployed, this should be described.

The description should also discuss whether the warden can be a single entity (centralized warden) or has to be multiple distributed instances (distributed warden), and whether the warden has to keep flow state (stateful warden) or can operate without flow state (stateless warden).

6.6 Literature Analysis

In this section, we provide an analysis of existing publications that describe hiding methods. The goal of our analysis is to determine how well the attributes of our unified description method are described within these existing papers.

To identify relevant publications, we selected those papers that appeared in a Google Scholar search using the search terms 'network AND covert channel(s)' and 'network steganography' as well as the papers that were cited in (Wendzel et al., 2015). For each publication that we found, we made sure that it described at least one network information hiding method. We limited our search range to the years 1987-2015 as the first academic work on network information hiding was published in 1987. As a result, we analyzed 74 publications that presented new hiding methods. We made the list of all analyzed papers available under the URL `http://www.wendzel.de/J.UCS/db_publications.pdf`.

Figure 6.4 shows the analyzed publications per year. In early years, only few papers on network covert channels were published. The number of papers grew over time due to the increasing popularity of the topic. As some papers describe more than one hiding technique, the number of analyzed hiding techniques (131 in sum) is sometimes larger than the number of publications, which is also shown in Figure 6.4. The number of found publications in the year 2015 is comparable low due to the delayed indexing of academic publications.

As can also be seen in Figure 6.4, there is a peak of hiding techniques published in the year 2006. This peak is a result of the publication (Lucena et al., 2006) which presented 22 hiding techniques for IPv6, AH and ESP in a single document.

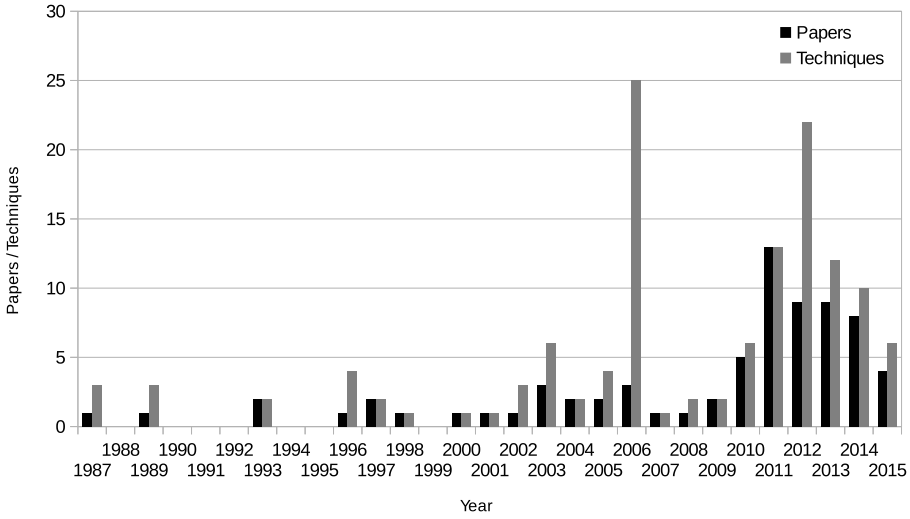

FIGURE 6.4: Analyzed publications that present hiding methods (per year).

Finding 1: Several papers lack fundamental attributes

As already stated in the introduction, our analysis shows that publications on hiding methods present varying subsets of attributes. Figure 6.5 provides an overview of the present attributes for all described hiding methods of the analyzed papers. When an attribute's description is classified as 'partial', the authors provided some aspects but lack other fundamental aspects of the particular attribute. The comprehensive description of an attribute was marked with 'yes' (fully present).

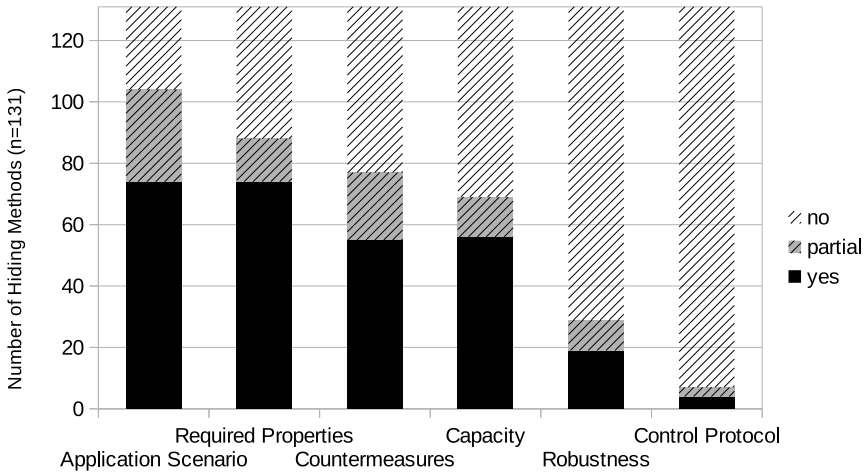

FIGURE 6.5: Presence (fully or partially) of selected attributes in the publications.

Out of the 131 described hiding methods, the application scenario was provided for 78% of them (74 fully, 30 partially). The required properties for the hiding method were fully described for 74 and partially for 14 hiding methods (combined 67%). For some of the techniques, the authors provided countermeasures: 55 contained a full description with evaluation of at least one countermeasure and for another 13 techniques, possible countermeasures were at least briefly discussed (combined 58%).

The channel capacity was evaluated for 52% of the hiding methods (56 fully, 13 partially; both values also including throughput and bitrate measurements). The robustness of the proposed hiding methods was discussed for only 22% of the hiding methods (19 fully, 10 partially). Control protocols are not part of most hiding methods and for this reason only described for 5% of the hiding methods (4 fully, 3 partially).

Finding 2: Attribute coverage changed over time

The attributes covered by publications changed over time. Figure 6.6 provides an overview of selected attributes over time. We omitted the sender-side and receiver-side processes that were described in most publications but in very varying detail.

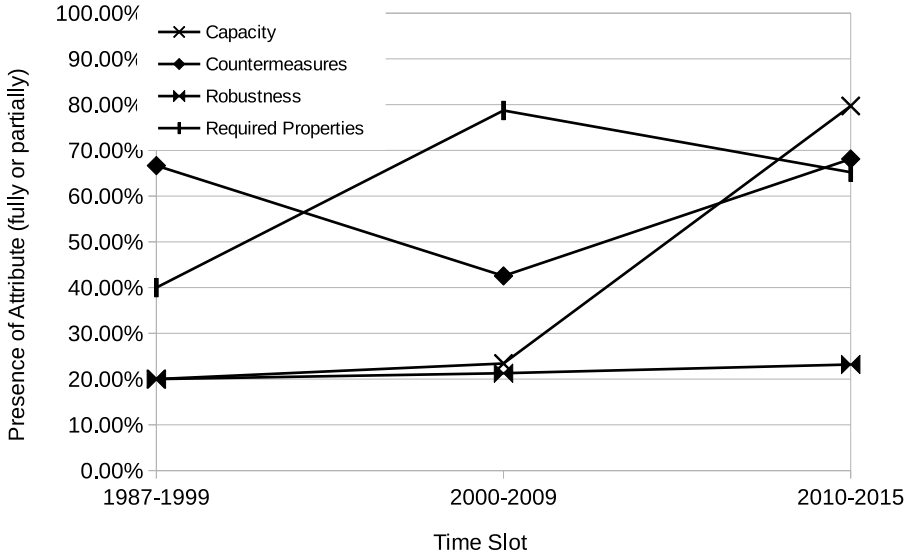

FIGURE 6.6: Coverage of selected attributes for hiding methods over time.

Over time, the covert channel capacity was increasingly discussed, especially in publications from the last six years (2010–2015). Channel robustness was constantly discussed for only 20% of the hiding methods. Both, channel capacity and robustness are part of the 'Covert Channel Properties' in our description method. The discussion of countermeasures varied over time and is similar in the range 2010–2015 as it was in 1987–1999 (approx. 68%). The required properties of the carrier were discussed by fewer publications in the years (2010–2015) compared to the years 2000–2009.

Finding 3: Domination of few hiding patterns

Hiding patterns were proposed only a four years ago (2015). For this reason, none of the analyzed publications covered any hiding patterns. We analyzed all hiding methods by verifying whether they were already assigned a pattern in Wendzel et al., 2015 and for several methods which had not been analyzed in that publication, we determined their hiding patterns. We were able to assign 130 of the 131 analyzed hiding methods to their respective patterns. One publication discussed a steganographic key exchange that applies to many hiding methods and thus cannot be assigned to a hiding pattern. Figure 6.7 shows the distribution of hiding patterns. As shown, most of the hiding methods we found were categorized as 'Value Modulation', followed by the 'Inter-arrival Time' pattern. For the patterns 'PDU Order', 'Re-Transmission' and 'Rate', less than five hiding methods were found. These findings can support the development of countermeasures as an efficient countermeasure may target one of the predominant hiding methods instead of a hiding pattern that is barely implemented.

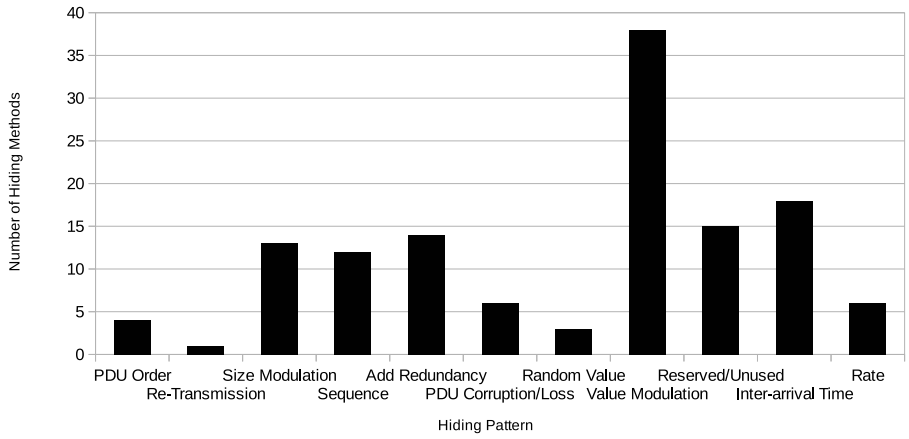

FIGURE 6.7: Occurrences of hiding patterns for the analyzed hiding methods.

Finding 4: Varying countermeasure descriptions

We found that the coverage of countermeasures for proposed hiding methods varies greatly among the analyzed papers. While several papers highlight their countermeasures briefly (sometimes only within a single sentence) other publications discuss them in more detail.

We analyzed whether papers covered one of the three types of countermeasures: detection & auditing, limitation, or prevention. Of those papers which provide a more-detailled discussion of potential or evaluated countermeasures, most cover a detection or prevention approach while very few discuss auditing or limitation approaches – examples of the latter are Girling [1987] (auditing and limitation) and Sadeghi et al. [2012] (limitation and prevention). Some works do also cover several countermeasures of the same type, e.g. Cabuk et al. in [2004] propose three methods for the detection the Inter-packet timing method (Section 6.7.1).

Overall, we found that the vast majority of publications do not discuss all types of countermeasures or even two of them in detail, i.e. featuring an actual evaluation of these. We believe that this is the most significant drawback of current research. However, it must be empasized that researchers are expected to publish several papers during their carrer while facing time pressure. For this reason, countermeasures are sometimes discussed in one or multiple follow-up papers. This approach, however, makes it more difficult to track the progress on research for a particular hiding method.

Finding 5: Inconsistent method descriptions within papers

Some of the publications describe several hiding methods while applying an inconsistent description for these within the same work. We use (Tuptuk and Hailes, 2015) as an example for this scenario. The authors present two covert channels for pervasive computing environments. The first signals hidden information by modulating the Link Quality Indication (LQI) of 802.15.4 wireless networks while the second modulates the values of a temperature sensor to signal hidden information. Table 6.1 indicates which of the selected attributes of the description are present, partially present, or not present. The table reveals that different attributes are described for each channel. The lack of a unified description makes it harder to compare different channels.

Hiding Method	Application Scenario	Required Properties	Countermeasures	Sender-Receiver Relation	Direct/ Indirect	Robustness	Capacity
Link quality (A)	●,☑	◐	●	○	○	●	●
Sensor data (A)	●,☑	◐	◐	●	◐	●	◐
SDP o-tag (B)	●,☑	◐,☑	○	○	○	◐,☑	◐,☑
SDP a-tag (B)	●,☑	◐,☑	○	○	○	◐,☑	◐,☑

TABLE 6.1: Presence of attributes in descriptions (●= present, ◐= partly present, ○= missing, ☑= combined description). Exemplified using Tuptuk and Hailes, 2015 (**A**) and Tsiatsikas et al., 2015 (**B**)

However, the abovementioned paper is not the only paper containing an inconsistent method description as several other papers contain similar descriptions. We assume that inconsistent descriptions of hiding methods are also linked to the necessity to shorten papers to match certain page limits of workshops and conferences. Such page limits can be matched easier when descriptions are combined for multiple hiding methods.

Finding 6: Combined evaluation of multiple hiding methods

We also found that sometimes multiple hiding methods are treated in a combined way throughout a paper. Table 6.1 summarizes the combined and the non-combined attributes of (Tsiatsikas et al., 2015). The authors describe two hiding methods for SIP useful for the stealthy command-and-control communication in a botnet. The first channel uses the mandatory '< o >' tag to carry hidden information while the second uses the optional '< a >' tag and its parameter to do the same. Both channels are combined

for the evaluation process as both channels are necessary to transfer the required secret message. This combined evaluation does not allow the reader to understand the performance of each channel separately and it also makes the comparison against other methods difficult.

Finding 7: Varying scenario descriptions

The standard scenario used in information hiding is Simmon's *Prisoner's Problem*, see Chapter 2.6. A variation of this scenario is the case where the communication channel is not mediated by the warden (Carrara and Adams, 2016).

The Prisoner's Problem is used by several publications, especially early ones such as Handel et al. [1996]. However, even early publications provide a variety of scenarios. For instance, Kang and Moskowitz discusses network covert channels in the context of multi-level security (MLS) (Kang and Moskowitz, 1993), i.e. the focus is on a policy-breaking communication and not on a stealthy communication. The scenario by Rowland is the communication of a trojan horse (Rowland, 1997) and in (Wolf, 1989), the author provides the scenario of bypassing filters.

Ahsan and Kundur describe a version of the Prisoner's Problem in a network scenario (Ahsan and Kundur, 2002a). Alice and Bob are connected to a LAN with their workstations. Both wish to establish a stealthy communication but a security-aware warden (network administrator) is present. Similarly, Wendzel et al., 2012 discuss the scenario of a cyber-physical-system (CPS), in particular a smart building. Using the smart building, two parties wish to establish a stealthy communication or use a (relatively insecure) smart building to exfiltrate data out of a cooperative network (Wendzel et al., 2014a). Alice and Bob therefore embed hidden information in a communicaton protocol used to transfer building automation-related data. Tuptuk and Hailes, 2015 discuss the scenario of pervasive computing in which a covert communication is performed. The use of a covert channel in their view is either to leak potentially sensitive information or to influence a CPS' operation.

Malware communication is another scenario domain. Tsiatsikas et al., 2015 use the scenario of a stealthy command & control communication for botnets. Calhoun et al. mention two different scenarios: The first is a covert authentication while the second is a wifi-version of the already mentioned botnet that uses a covert channel for its command & control communication (Calhoun Jr et al., 2012).

Although Craver's work [1998] is not specific to network communications, it discusses a public-private key-based approach for steganographic communication, e.g. in audio or video clips. Key exchange can be seen as a fundamental aspect for the communication scenarios as it must be performed a priori in order to allow any stealthy communication between Alice and Bob.

6.7 Exemplary Descriptions

We now present two exemplary descriptions. The first description is for a covert timing channel, while the second description is for a covert storage channel.

6.7.1 Example 1: Inter-packet Timing Method

In this example we describe a specific steganographic method for hiding information in inter-packet timings. This method or channel is also referred to as model-based inter-packet gap channel and was proposed by Gianvecchio et al., 2008.

Hiding Pattern

As the covert signaling utilizes the timing of network packets, the method belongs to the 'Network Covert Timing Channel' pattern. In particular, the method falls under 'Inter-arrival Time'. The full path in the pattern hierarchy is:

```
Network Covert Timing Channels
`-- Inter-arrival Time Pattern
```

Application Scenario

The method can be used for general-purpose covert communication between a covert sender and a covert receiver or between a group of covert parties depending on whether the carrier is unicast or multicast.

Properties of the Carrier

The method only requires that the carrier consists of packetized data, such as network-layer packets, whose timing can be manipulated. The method assumes that there is sufficient noise in the timing of packets by senders and along the path, so that manipulations of timings are not immediately suspicious. While the method was proposed and evaluated for HTTP in (Gianvecchio et al., 2008), it is not limited only to this protocol. However, some carrier protocols are more suitable than others. Since the encoding destroys any dependence between the inter-packet times of successive packets, it is best used with carriers that already have independent inter-packet times (Zander et al., 2011).

Sender-side Process

The embedding process involves fitting a model to the inter-packet time distribution of regular traffic and then using the model to generate covert

channels with identical distribution (for details see (Gianvecchio et al., 2008)). There is usually a single sender process that embeds the covert channel in a single carrier. Note that a single carrier can be multiple traffic flows.

Since the carrier is HTTP/TCP, the reliability of TCP (handshake, teardown, sequence numbers, ACKs) provides basic bit synchronization and reliable in-order delivery of bits for a single carrier flow. A fully reliable channel, supporting multiple carrier flows, would also require the framing of messages and frame synchronization. While (Gianvecchio et al., 2008) did not discuss this, existing techniques could be used.

In the original work the covert sender generated the overt traffic (Gianvecchio et al., 2008). However, the method can be also applied to embed the covert channel into existing network traffic at the cost of increasing the latency of the overt traffic (Zander et al., 2011).

Receiver-side Process

The extracting process running on the single receiver decodes the covert bits from the observed inter-packet times of the single carrier as described in (Gianvecchio et al., 2008).

The receiver can leverage TCP's reliability, so there are no lost or reordered bits for a single TCP carrier flow. Additional higher-layer mechanisms may be needed for full reliability as discussed in Section 6.7.1.

Covert Channel Properties

The method can be used in the end-to-end scenario, where covert sender and receiver are also the overt sender and receiver, and in a MitM scenario, where covert sender and receiver are placed between the actual sender and receiver (as well as in hybrid scenarios). The method creates a direct channel between covert sender and receiver(s). The steganographic bandwidth depends on the channel noise and the packet rate of the carrier traffic. Gianvecchio *et al.* measured capacities of 5–20 bits per second in their experiments. Note that in practice the goodput is likely smaller as part of the capacity will be used by a control protocol to provide reliable transport. The channel is hard to detect only if the regular traffic has uncorrelated inter-packet times which is largely the case when HTTP is used as a carrier (as in (Gianvecchio et al., 2008)). Otherwise the channel can be detected with metrics that can measure the dependency of inter-packet times (Zander et al., 2011). The channel is robust against typical network packet timing noise. If an active warden can manipulate packet timings without impunity, the capacity of the channel would be severely reduced up to a degree where the channel would be practically eliminated. Measurements regarding the steganographic cost were not provided by the authors as the concept of steganographic cost had not been introduced at that time. The steganographic cost depends on the abovementioned channel characteristics and the carrier traffic. In general, the more severely delays are perturbed in overt traffic, the higher the steganographic cost.

Control Protocol

Gianvecchio et al., 2008 only describe the "physical layer" of the covert channel (encoding/decoding of bits) and do not mention a control protocol.

Countermeasures

The covert channel can be limited or even practically eliminated by introducing timing noise, either at the sender or in the network (Fisk et al., 2003). Depending on the carrier this may introduce unwanted side-effects though, for example, it may add additional latency to the carrier application's traffic.

The covert channel mimics the distribution of inter-packet times of the normal traffic. This makes the channel hard to detect if the normal traffic has uncorrelated inter-packet times (Gianvecchio et al., 2008). However, for applications that have correlated inter-packet times, the channel can be detected with metrics that can measure the dependency of inter-packet times (Zander et al., 2011).

6.7.2 Example 2: DHCP Number of Options Storage Method

We now discuss the description of a covert storage channel. Rios *et al.* presented several DHCP-based covert channels, of which one hides information by changing the number of DHCP options in a DHCP packet (Rios et al., 2012).

Hiding Pattern

As the modification of DHCP options represents the modification of a storage attribute, the method falls under 'Network Covert Storage Channels'. The DHCP options are part of the DHCP header and thus, a 'Modification of Non-Payload'. They are also 'Structure Modifying' as the header structure is extended when DHCP options are embedded. The signaling of the hidden information is performed in a way that a sequence of objects (DHCP options) is utilized ('Sequence Pattern') and, in particular, the number of options represents the hidden information itself ('Number of Elements Pattern'). The full path in the pattern hierarchy is:

```
Network Covert Storage Channels
'-- Modification of Non-Payload
    '-- Structure Modifying
        '-- Sequence Pattern
            '-- Number of Elements Pattern
```

Application Scenario

Rios *et al.* discuss a potential application in a data exfiltration scenario (Rios et al., 2012): Alice, having privileged access to an embassy network, needs

to receive information from Bob, but the direct communication between Alice and Bob is forbidden and the Internet-based communication between them would be suspicious. Bob visits the embassy and transfers network messages to Alice. He embeds hidden information within the non-blocked DHCP protocol using the local network. The application scenario foresees only an uni-directional communication, i.e. no backwards channel from Alice to Bob, but in general the channel could be bi-directional.

Properties of the Carrier

The DHCP protocol must be allowed, i.e. not administratively prohibited by a network security policy (e.g. blocked by a switch or layer-2 firewall). As the intended transfer of hidden information is uni-directional, i.e. only from Bob to Alice, Alice is not required to be able to send over the carrier herself. In addition, sender and receiver must be located within the same network.

The hiding method is protocol-specific and can only be applied to the DHCP protocol. To apply the technique, the network must not block particular DHCP options and the encoding of hidden information must be performed in a way that for all encodable symbols, the resulting DHCP packet is still transferable over the carrier.

Sender-side Process

The secret sender generates its own overt traffic. The sender-side process embeds a hidden symbol by adding the number of DHCP options the symbol requires for its encoding to the DHCP packet. At least two options must be embedded due to the DHCP standard, for example, if the symbols are 'A'–'Z', the symbol 'A' requires two options already (Rios et al., 2012). The symbol 'Z' would require 27 options, which is likely to raise suspicion (Rios et al., 2012) and may be blocked by firewalls. Each symbol to be transmitted must be encoded in a separate packet.

Reliability is not implemented directly – instead the *recovery mechanisms provided by DHCP against packet loss* are exploited (Rios et al., 2012).

Receiver-side Process

The receiver observes DHCP messages sent by the covert sender and counts the number of embedded DHCP options. The number of DHCP options represents the hidden symbol. The decoding is performed separately for each DHCP packet and the received symbols are combined to reassemble the transmitted message.

Covert Channel Properties

The method works in an end-to-end communication scenario. It cannot be used in a MitM scenario due to the properties of DHCP (broadcast messages that are limited to one subnet). The channel is a direct channel. The bandwidth of the channel depends on the number of DHCP packets sent per second. In general, the channel can transfer as many symbols as packets per second. The detectability of the channel increases with the number of symbols encoded per second and with the size of the encoded symbol. The detectability of a message transfer could be improved by encoding the most frequently used symbols with shorter messages (i.e. apply Huffman coding). Robustness of the covert communication is provided by the use of DHCP's recovery mechanisms. Measurements regarding the steganographic cost were not provided by the authors as the concept of steganographic cost had not been introduced at that time. In general, the distortion of the used carrier (e.g. the Ethernet environment) is minimal as long as the number of DHCP packets does not influence the network's performance in a recognizable manner.

Control Protocol

No control protocol was described in (Rios et al., 2012). As stated in (Rios et al., 2012), a bi-directional communication over the covert channel is feasible, so bi-directional control protocols could be integrated.

Countermeasures

The easiest way to eliminate the method is to prevent the use of the DHCP protocol or DHCP packets with more than two options. However, this solution may not be practically applicable as the DHCP protocol is essential in most networking environments. A traffic normalizer that deletes uncommon or redundant DHCP options would be a better solution but it may eliminate actually required protocol functionality.

Another potential countermeasure would be to limit the number of DHCP packets per second. This approach reduces the channel's performance as each symbol must be encoded in a separate packet. As the number of DHCP packets per second is unlikely to be high during regular transmissions, a statistical analysis will probably allow an accurate detection of the steganographic method.

Rios *et al.* state that large DHCP packets, i.e. those with many options, may raise suspicion (Rios et al., 2012). DHCP packets with an unusual large number of embedded options can likely be detected with simple intrusion detection rules.

6.8 Linking Description Method and Creativity Framework

In the creativity framework (Wendzel and Palmer, 2015), the major focus is on the evaluation of creativity, especially originality, of a proposed hiding method. We will briefly describe the creativity framework and afterwards explain how the new unified description method can be used to improve it.

Steps of the Creativity Framework: The creativity framework consists of five steps which are aligned with the traditional peer review process. In step one, a pattern database is generated by the research community. Due to the availability of an existing pattern catalog, step one is already accomplished.

In the second step, the authors create the idea of a new hiding method, e.g. to embedd hidden bits into a new network protocol. The authors describe their new hiding method in form of a scientific paper in step three. They justify the novelty of their method using the *creativity metric*, which is based on the originality of the method (hiding method pattern) as well as the context (application scenario and carrier network protocol).

The authors submit their paper to a peer review. The reviewers evaluate the novelty of the work using the creativity metric and decide whether the proposed work is a "Big-C" or a "small-c" contribution, i.e. whether the work consists a high level of creativity. Big-C and small-c are standard terms from creativity research. Only in the Big-C case, the work is accepted to represent a new pattern and its pattern description is optimized in step four, otherwise step four is skipped.

The work is published in step five. In case of "Big-C" the publication has to state that the work represents a new pattern – this automatically extends the pattern database. In case of "small-c", the hiding method is published but the authors cannot claim that they have discovered a new pattern; instead they provide new results for an existing hiding pattern.

Benefits of Combining Both Approaches: The creativity metric does not enforce a detailed description structure. Our unified description method can replace the descriptive aspects of the creativity metric since it provides a more fine-grained description and allows for the distinction and comparison of various aspects of a hiding method. Several attributes, such as whether a hiding method can operate in a MitM setup or a distinction between the sender-side and receiver-side processes, were not covered in (Wendzel and Palmer, 2015). On the other hand, our unified description method can benefit from the creativity framework. There is no reason to create a new approach for integrating the unified description method into the peer review process. Also, the creativity framework evaluates the novelty of a hiding method by applying research from creativity psychology, which can serve as an add-on to the technical evaluation of the unified description method. In summary, the combination of both approaches can be considered beneficial.

6.9 Discussion and Conclusion

We developed an approach to unify and structure the description of network steganographic methods. To this end, we performed a comprehensive literature analysis in the domain to identify requirements for the description method. Currently, no such description exists, making it difficult to compare the published work on hiding methods. Unified descriptions of hiding methods are desirable as they ease the comparison of research results, supporting the scientific efficiency in the research domain (cf. Chapter 3.4). They also improve the accessibility of hiding methods and foster the reproduction of experimental results. As a key aspect of our description method is the association of hiding methods with a hiding pattern, the approach automatically enforces a categorization of the research results into the known pattern taxonomy. Last but not least, scientists can easily spot research gaps as they see what aspects of the method were already documented (e.g. channel capacity or detection methods) and which aspects are left for future work.

However, it could be difficult to win recognition for a unified hiding method description as, in the end, it is up to every author to decide how to describe his or her hiding method. For this reason, we designed our description so that it will be applicable and attractive for hopefully many authors. Our description structure does not specify every single detail of all attributes. It leaves several decisions to the authors, such as whether or not to discuss details of certain attributes (e.g. covert channel capacity), the form of descriptions (e.g. text or figures), and the extent of the descriptions. This flexibility allows to apply the unified description method also in short papers which are, for many conferences, limited to four to six pages. As one attribute is specified as 'optional', it can be also left out.

We identified a benefit when combining the new unified description method with the existing 'creativity framework'. The framework's process is kept but the 'creativity metric' of the framework is replaced with our unified description method.

Chapter 7

Adressing the Lack of Experimental Replications

Abstract While the previous chapters were focusing on the taxonomy, evaluation and description of hiding methods, this chapter will provide the next step for the advancement of the methodology in the domain. In particular does this chapter address the problem of replication studies.

Therefore, this chapter presents WoDiCoF, a distributed testbed, accessible for the international research community to perform a unified evaluation of detection algorithms for network covert channels. In comparison to existing works, our testbed is designed for upcoming big-data scenarios, in which huge traffic recordings must be analyzed for covert channels. It is the first testbed to allow the testing of parallel detection algorithms.
To evaluate WoDiCoF, we took a detection algorithm published in ACM CCS/TISSEC, verified several of the original results and enhanced the understanding of its performance by considering previously unconsidered parameters. By parallelizing the algorithm, we could moreover achieve a speed-up of 2.89 with three nodes.

Originally published: Keidel, Wendzel, Zillien, Conner, and Haas (2018)

7.1 Introduction

Previous chapters and related publications discussed the scientific fundamentals of network covert channels, e.g. cf. (Wendzel et al., 2016; Mazurczyk et al., 2016a; Chen et al., 2017; Wendzel et al., 2017b), and several publications address the scientific fundamentals of steganography itself, e.g. (Katzenbeisser and Petitcolas, 2002; Anderson and Petitcolas, 1998; Anderson, 1996). However, none of these publications addresses a central problem of covert channel research: network covert channel detection algorithms are not evaluated in a way that fosters reproducible experiments similarly as it is common practice in natural sciences. An additional problem that is covered in this chapter, too, is that network covert channel detection methods are not designed for parallel algorithms.

In this chapter, we present WoDiCoF (*Worms Distributed Covert Channel Detection Framework*), a testbed that improves upon existing scientific methodology in network covert channels in the following ways:

- it allows the evaluation of parallel network covert channel detection algorithms (and thus also aids big-data analysis);
- is remotely accessible for the scientific community;
- researchers can provide code and configuration files of their detection algorithms and covert channel techniques, which enables reproducible experiments under specified conditions;
- provides a traffic generator that is tailored for network covert channel analysis.

To illustrate the capabilities of WoDiCoF, we moreover enhance the current understanding of a well-cited covert channel detection algorithm published by (Cabuk et al., 2009) in CCS and TISSEC. We additionally parallelized the algorithm to achieve a speed-up.

WoDiCoF was designed and implemented by the *Center for Technology and Transfer* (ZTT) of the University of Applied Sciences in Worms and is additionally a project that runs under the umbrella of the CUING initiative[1].

The remainder of this chapter is structured as follows. In Section 7.2, we discuss related work. We present the design and the implementation of our testbed in Section 7.3 while Section 7.4 shows the evaluation of a sample covert channel detection using a well-known algorithm by Cabuk et al., for which we present enhanced insights. Potential drawbacks and limitations of WoDiCoF are discussed in Section 7.5 while Section 7.6 concludes and provides an outlook on future work.

7.2 Related Work

Several testbeds for security research and testing have been presented. For instance, Benzel et al. designed and configured DETER (Benzel et al., 2007),

[1]CUING (*Criminal Use of Information Hiding*) is an initiative supported by Europol's European Cybercrime Centre (EC3) cf. `http://www.cuing.org`

a testbed comprised of hardware in combination with extensive control software to experimentally verify the effectiveness of attacks and defenses of malicious code. Siaterlis et al. designed and configured EPIC (Siaterlis et al., 2013), a testbed specifically designed for cyber security studies with multiple heterogeneous Networked Critical Infrastructure

However, only few testbeds on network covert channels have been presented so far. Zander developed CCHEF, the *Covert Channels Evaluation Framework*, cf. (Zander and Armitage, 2008). CCHEF is an extensible tool that allows the creation of covert channels for computer networks. With CCHEF, traffic can be embedded using traffic recordings. CCHEF is set-up using configuration files, which would allow for reperforming conducted experiments.

Zseby et al. developed a network steganography testbed for higher education at the Technical University of Vienna (Zseby et al., 2016). Their testbed involves CCHEF as a tool for covert channel creation and evaluation but also allows teaching students the statistical analysis of abnormal traffic patterns.

In November 2017, Gunadi and Zander developed an extension for the *Bro* intrusion detection system. Their extension allows for detecting network covert channels (Gunadi and Zander, 2017b). The authors utilize the concept of 'hiding patterns' that were introduced by (Wendzel et al., 2015) and work on the integration of detection modules for all known patterns. Implementation details are provided in (Gunadi and Zander, 2017a) specifically regarding the applied analysis methods: Kolmogorov-Smirnov test (KS test), entropy and corrected conditional entropy as well as multi modal analysis.

However, none of these testbeds address all of our major design goals, especially the support of parallel detection algorithms and the remote accessibility of the testbed by default.

In Chapter 3, we highlight the need for experimental verification by re-executing experiments described in papers with data, parameters and tools of the particular authors. WoDiCoF aids this process by design.

7.3 Design & Implementation of WoDiCoF

In this section, we first provide an overview of WoDiCoF's design concept, followed by a detailed discussion of its implementation.

7.3.1 System Requirements

The decision for the open source framework *Apache Hadoop*[2] as our technology platform is based on the capability to support performance and memory space scaling by adding commodity hardware to our cluster or by simply renting cloud computing resources. Additionally it is our goal to offer

[2] cf. `https://hadoop.apache.org`

low barriers to entry for experimentation and analysis. The ease of actually performing parallel tasks as well as no requirements on the development language beyond supporting standard input/output are crucial benefits in this regard. Finally, although there is a certain up-front administrative cost to running a Hadoop based system, we have learned that once successfully installed, only a minimum of care is needed.

Graphics processing unit (GPU) based systems might be faster for special applications, but we are bound to support a wide variety of as yet unknown detection algorithms. Some of these, such as ones based on breadth first or depth first traversals would perform extremely poorly on GPU based systems. We are confident that our approach offers good trade-offs regarding general applicability and performance in a wide range of scenarios.

7.3.2 Design Concept Overview

The design of WoDiCoF contains several components as depicted in Figure 7.1. Traffic can either be generated using a traffic generator or by utilizing previously recorded traffic. Traffic recordings are accepted as PCAP and legacy ethernet format files (from NZIX, *New Zealand Internet Exchange*).

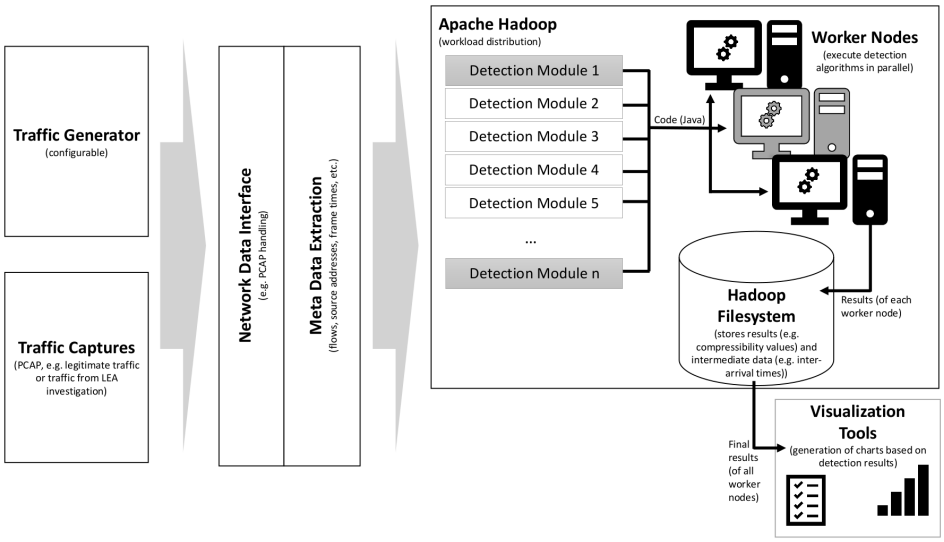

FIGURE 7.1: Overview of WoDiCoF.

Following input, meta information is extracted from the traffic recordings. This includes: timing values of recorded frames, source and destination addresses, employed protocols, source and destination ports and packet sizes. This information is then provided to an Apache Hadoop-based system for parallel preprocessing and analysis.

In the Hadoop system, we sort and group the data into flows, derive flow-dependent *inter-arrival times* (IATs)[3] from the relative timing values of the recorded frames and hand these off for analysis. Large flows can be further split in advance into small windows (e.g. $N = 2,048$ packets) to better

[3]IATs represent the elapsed time between two consecutive network packets of one or multiple flows. IATs are sometimes also called *inter-packet gaps* (IPGs).

leverage parallelization advantages. Apache Hadoop performs the parallel execution of the covert channel detection algorithms based on the flow data. Such detection algorithms can be added by researchers using a Java-based interface.

Finally, results are provided on a *Hadoop Distributed Filesystem*[4] (HDFS) where they can be processed further by traditional tools, such as *Gnuplot* and Python libraries, to produce figures to aid the academic paper writing process.

7.3.3 Implementation Details

While the previous section provided a high-level overview of WoDiCoF, we will now highlight details of our implementation.

Traffic Generator

The main purpose of our traffic generator is to create highly configurable synthetic network traffic with specific statistical characteristics.

While several tools for traffic generation exist, e.g. (Haas, 2010; Garcia and Fyodor, 2017; Klauser, Borkmann, et al., 2017; Stefano Avallone and Ventre, 2003), all of these fail to *combine* both, flexible covert channel creation (e.g. choosing among different methods to manipulate header bits for information hiding) and scientific applicability (e.g. defining value distributions for header fields or packet timing in a way that is typical for covert channels). For this reason, WoDiCoF contains its own tailored traffic generator that allows both: the generation of covert channel traffic *and* the typical options of other traffic generators, i.e. to define a detailed configuration of several traffic characteristics, such as packet count and distribution of certain values. Our traffic generator is written in Python and uses YAML as a configuration markup language. It utilizes the Scapy packet generator[5] internally to provide a broad configuration interface and support for multiple protocols (TCP, UDP and ICMP).

For each protocol there are a number of configuration parameters available. In the case of TCP, it is possible to simulate a handshake and answers with acknowledgements while the TCP flags and IP options can be manipulated. For ICMP, it is also possible to simulate responses and to manipulate the IP options. For UDP, is it again possible to manipulate the IP options. The number of generated packets can be defined for all protocols.

As mentioned, a central functionality is the manipulation of IATs between packets. The IATs can be generated using a variety of statistical distributions or can be drawn from a user defined list with given probabilities. It is also possible to add another layer of noise on top of the IATs to simulate network jitter. This gives the user several options to simulate legitimate

[4] cf. HDFS Architecture Guide `https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html`.

[5] cf. `http://secdev.org/projects/scapy/`

traffic and covert channels. To make the generated traffic reproducible, all used PRNGs are seeded and the seed can be defined in the configuration.

Network Data Interface and Metadata Extraction

The traffic recording input is converted into a comma separated value file (CSV) format that contains only the metadata that is relevant for the detection algorithms. The conversion is done using scripts. The CSV input data is then provided to the Apache Hadoop framework.

Utilization and Functionality of Apache Hadoop

The Apache Hadoop Library allows for the distributed parallel sorting and processing of arbitrary data on commodity systems. Possible clusters can be composed of a few or of up to thousands of nodes. Hadoop employs the MapReduce programming model (Dean and Ghemawat, 2004), a distributed parallel data processing paradigm inspired by functional programming concepts. This enables sorting and processing of suitable big data with the benefit of reduced overall computation time.

The MapReduce programming model consists, unsurprisingly, of two stages: the mapper and the reducer, both of which are parallelized. We use the mapper stage to sort and group the input data into flows. A flow determines an IP-conversation between two endpoints. The covert channel analysis as well as some preprocessing, such as computing IATs from frame times, is performed in the reducer stage.

Each stage in the Hadoop system expects data to be provided as key/value pairs. In our mapper stage, the key is a tuple with the following structure: [*PCAP input filename, protocol name, IP source address, IP destination address, source port (if applicable), destination port (if applicable), frame time*] which together define a flow. The value is a tuple with the following structure: [*frame time, packet number, ...*]. The value tuple is used for the actual covert channel detection in the reducer stage and can be extended and modified as needed to support different analysis methods. The frame time is part of the key as well as part of the value to make it possible to implement a Secondary Sort Algorithm (Gunarathne et al., 2015), which enables the reducer stage to provide IP-conversations with a list of chronologically sorted values. Arbitrary numbers of modules can be called in the reducer to process the data. The results are stored in CSV formatted files on the HDFS volume.

The input data are CSV files which are easy to split by the Hadoop framework to achieve best results in terms of overall computation time. HDFS distributes the input data within the cluster. Both mapper and reducer processes are replicated and distributed over the cluster. This setup works well using a default configuration but can be tuned further to achieve optimal results. Overall computation time can be adjusted by changing the cluster size.

Visualization

After the detection algorithms process the traffic metadata using Hadoop, all results are aggregated from files residing on HDFS to CSV formatted files on a Linux filesystem. We apply scripts to process these results to generate visualization output. This is currently done using shell scripts (`bash`, `awk`, `sed` and similar tools) as well as `gnuplot`, `weka`[6] and `Python`. By default, plots are generated for all given parameters and for each flow, resulting in Gigabytes of output plots, depending on the size of the input traffic and the number of flows. However, a pre-selection is possible (e.g. by defining thresholds for certain parameters in the scripts).

The usage of scripts in combination with the mentioned UNIX tools and frameworks gives rapid results but is error-prone and cumbersome if parameters have to be adjusted. We are currently evaluating approaches to offer user interfaces for parameter adjustments which are self-explanatory and workflows to speed up the generation of plots for immediate visual results. We are targeting a professional audience with limited programming skills as well as an academic audience.

7.4 Evaluation of a Sample Approach

We evaluated the functionality of WoDiCoF by implementing a well-known detection algorithm presented by (Cabuk et al., 2004; Cabuk et al., 2009). The paper was initially published in ACM CCS 2004. In 2009, an extended version of the paper was published in ACM Trans. Inf. Syst. Security (TISSEC). In sum, Google Scholar reveals more than 500 citations of these publicaitons.

In the following, we first introduce the selected detection algorithm by Cabuk et al. and then explain how we extended the evaluation of their algorithm in comparison to the original work of the authors using WoDiCoF and compare our evaluation results with those of Cabuk et al. To the best of our knowledge, this is the first re-evaluation of a major covert channel detection algorithm. Enabling such re-evaluations (or proofs), as proposed in (Wendzel et al., 2017b), was a major goal of WoDiCoF.

7.4.1 Description of the Used Detection Approach

The underling scenario of Cabuk et al. are storage and timing channels that transfer data in the following ways:[7]

1. *storage channel:* either send data within pre-defined time-frames of duration τ, or not (indicating a zero or one bit, respectively).[8]

[6] `https://www.cs.waikato.ac.nz/ml/weka/`

[7] In the *original paper* by (Cabuk et al., 2009), this approach is described in Section 4.1.2.

[8] Note that this understanding of a storage channel is not perfectly aligned with the typical representation of a network covert storage channel as e.g. defined in (Mazurczyk et al., 2016a). However, in order to stay within the terminology of the original paper, we apply the distinction used by Cabuk et al. in the remainder of this chapter.

2. *timing channel:* Hidden data is encoded in IATs where the IAT τ encodes a zero bit and the IAT 2τ encodes a one bit.

The detection algorithm operates as follows: First, the IATs between network packets $\mathrm{IAT}_1 \ldots \mathrm{IAT}_n$ are recorded for every flow. IAT values > 1 are dropped to reduce noise. Secondly, the IATs are encoded into strings using a function which we call E. E first rounds the IAT to the first two significant digits behind leading zeros. E also encodes zero digits as an alphabetical character. For instance, the authors translate the IAT 0.00247s to 'B25' ('B' represents two zeros) and 0.0247s to 'A25' ('A' represents one zero). All string-encoded IATs of a flow are afterwards concatenated to a string S, i.e.

$$S \quad = \quad E(\mathrm{IAT}_1) \parallel E(\mathrm{IAT}_2) \parallel \ldots \parallel E(\mathrm{IAT}_n),$$

with $\parallel$ representing a string concatenation.

Thirdly, the authors approximate the Kolmogorov complexity of S, which is the maximum compression of a string, by compressing the string with the *gzip* algorithm, i.e. $C = \mathrm{gzip}(S)$, and then calculating the *compressibility* (compression rate):

$$\kappa(S) \quad = \quad \frac{|S|}{|C|},$$

with $|x|$ representing the length of the string x.

Cabuk et al. compare the compressibility of covert channel flows with non-covert channel flows. Their results indicate that covert channel flows and non-covert channel flows can be distinguished in specific cases that were presented in the original paper (we will discuss their results in Section 7.4.3).

Cabuk et al. transferred a textual string over their covert channel, using a connection between Purdue and Georgetown universities with τ set to 0.04, 0.06 and 0.08 sec. Their goal was to determine, whether the compressibility approach allows to *accurately detect a covert channel in a short window* (Cabuk et al., 2009). The authors used a window size of $N = 2,000$ packets. They performed a detection of covert channel traffic and compared their results with detectability results for legitimate traffic, including noise.

7.4.2 WoDiCoF-based Enhancement of the Analysis

In comparison to the original analysis by Cabuk et al., we did not reperform *all* of their conducted experiments. However, we extended the evaluation of their algorithm for the following aspects to underpin the capabilities of WoDiCoF:

1. We parallelized the algorithm to evaluate its performance depending on the number of nodes operating in parallel.
2. We implemented the coding of string S using different precisions of digits behind leading zeros of the IATs, i.e. not solely considering the first two significant digits behind leading zeros.

3. We evaluated the algorithm based on different content transferred over the covert channel. In particular, we transferred repetitive strings in the form (i) "ABABAB...", (ii) "The quick brown fox jumps over the lazy dog.", (iii) the first page of Goethe's *Faust*[9], as well as Faust compressed using (iv) GZIP, (v) ZIP, and (vi) BZIP2. No error-correcting codes were applied.
4. We analyzed covert channel traffic using different connections: a connection over local ethernet switches and a connection with a remote host in another autonomous system.

7.4.3 Evaluation Results

In the following, we provide our results for the parameters mentioned in the previous section and compare our results with results provided by Cabuk et al. (if available).

Parallel vs. Sequential Performance

One of the design goals of WoDiCoF was to enable parallel execution of detection algorithms using the Apache Hadoop framework. To achieve parallelization, input data are dynamically split by Hadoop and fed into the system. Hadoop automatically monitors the utilization of each node and ensures an equal workload distribution. As expected, this simple parallelization already achieves a speed-up as shown in Fig 7.2.[10] We calculated the speed-up S_n of the algorithm running on n nodes by dividing the serial execution time T_1 by the time the algorithm required to complete its execution on n nodes (T_n):

$$S_n = T_1/T_n$$

This calculation is analog to the classical speed-up calculation described in (Tanenbaum and Bos, 2015), where the theoretical speed-up is calculated depending on the available processors. However, in our case, T_1 and T_n were determined via time measurements.

We processed the NZIX-II recordings[11] that were used by Cabuk et al. and which contained 835 Mio. packets with the detection algorithm described above. If the code was executed on only one node, it took 3:05 hrs to process all data. When two nodes were used, the overall runtime was roughly halved (1:36 hrs, speed-up: 1.927). For three nodes, the computing time was 1:04 hrs (speed-up: 2.890), indicating small amount of serial code. However, these values solely include the parallel execution of the detection algorithm,

[9] The text of Faust is available under `http://gutenberg.spiegel.de/buch/-3664/4`.

[10] Please note that this flow-based workload assignment will not work for detection algorithms that need to consider data of multiple flows, cf. Section 7.5. Such algorithms are, for instance, required in the case of protocol switching covert channels (PSCC).

[11] `https://wand.net.nz/wits/nzix/2/`

not the pre-processing of the NZIX-II data, i.e., the initial extraction of metadata, which can take multiple minutes for such large traffic recordings.

All nodes were equipped with the similar resources (AMD Opteron 4180 or Intel Xeon E5603/E31260L CPUs, with eight cores each (running eight threads for Apache Hadoop), 4-8 GBytes of RAM, 1 GBit/s network link).

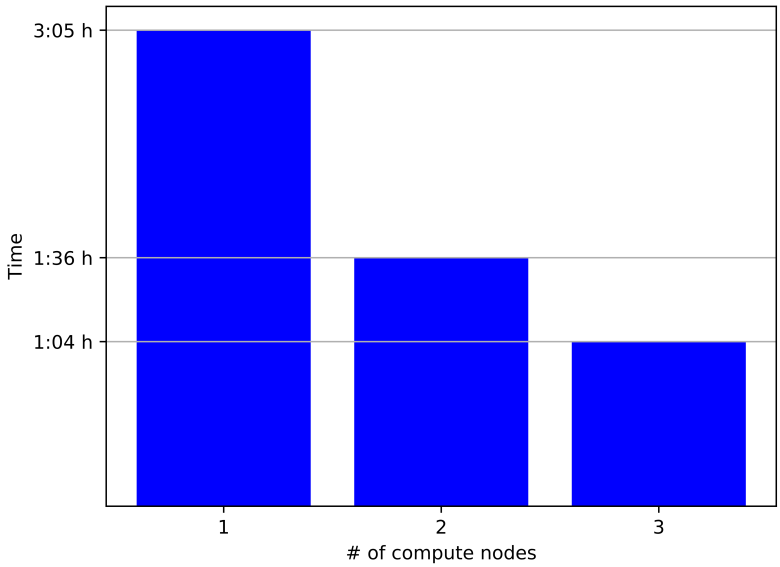

FIGURE 7.2: Comparison of computing time for compressibility-based detection depending on the number of nodes operating in parallel.

Compressibility-dependence on String Coding Precision

Cabuk et al. utilized a precision of two, i.e. the first two significant bits of the IAT where taken into account to perform a string compression. While Cabuk et al. selected only a few flows which all featured a κ value lower than those of their covert storage channels, our results in Figure 7.3 show that the precision (number of utilized relevant digits behind leading zeros) clearly influences κ. As also visible, the legitimate traffic in the NZIX-II recordings results in κ values that strongly overlap with the κ values of covert channel traffic for all tested protocols, i.e. legitimate UDP, TCP and ICMP traffic. However, for a higher precision, κ of legitimate traffic is smaller (below seven for precission 4). Moreover, storage covert channel (SCC) and timing covert channel (TCC) traffic overlap in their κ values.

Compressibility-dependence on Transferred Data Type

Cabuk et al. did not distinguish how the transferred data of the covert channel influences the compressibility (κ). As long as encrypted, random or compressed content is transferred, our compressibility results for storage channels are at least similar to those of Cabuk et al., i.e. our results partially

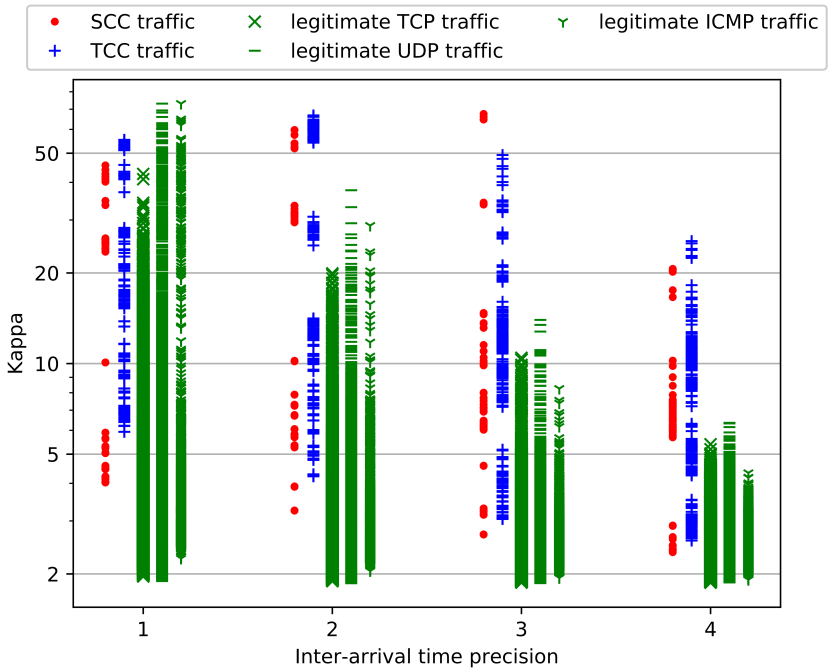

FIGURE 7.3: Dependence of κ on the type of applied precision and traffic type.

confirm the results of Cabuk et al. (see Figure 7.4). However, we also transferred different types of plaintext over the covert channels. In particular, we additionally transferred the text Faust, the repeating string "ABABAB...", the GZip, BZip2 and ZIP compressed version of Faust and the text "The quick brown fox jumps over the lazy dog" in loops over the channel. Each traffic was transferred using the τ values used by Cabuk et al. i.e. 0.04 sec, 0.06 sec, and 0.08 sec.

Figure 7.4 shows the results for a storage channel (SCC). For $\tau = 0.04$ sec, our results match those of Cabuk et al. if the data is compressed with ZIP or GZip. We achieve a lower compressibility for BZip2 and significantly larger compressibility scores for Faust and highly-compressible content ("ABABAB" and "The quick brown fox ...").

For $\tau = 0.06$ and for $\tau = 0.08$, we cannot confirm consistency with the result of Cabuk et al. in our experiment as the compressibility of Cabuk et al. matches approximately the compressibility of uncompressed the Faust text. For all compressed data, we achieve lower κ values.

The texts "ABABAB..." and "The quick brown fox" resulted in high compressibility scores for all tested values of τ.

Cabuk et al. observed that with an *increase [of] timing interval for IP SCC, compressibility increases*, too. Our results cannot confirm this observation as shown in the difference for our values and the values by Cabuk et al. in Figure 7.4. Instead, our compressibility results slightly decreased or remained stable. However, it must be noted, that we did not transfer exactly the same text that was used by Cabuk et al. (but several other texts) and used our own covert channel tool to generate the covert traffic (no error-correcting codes were applied).

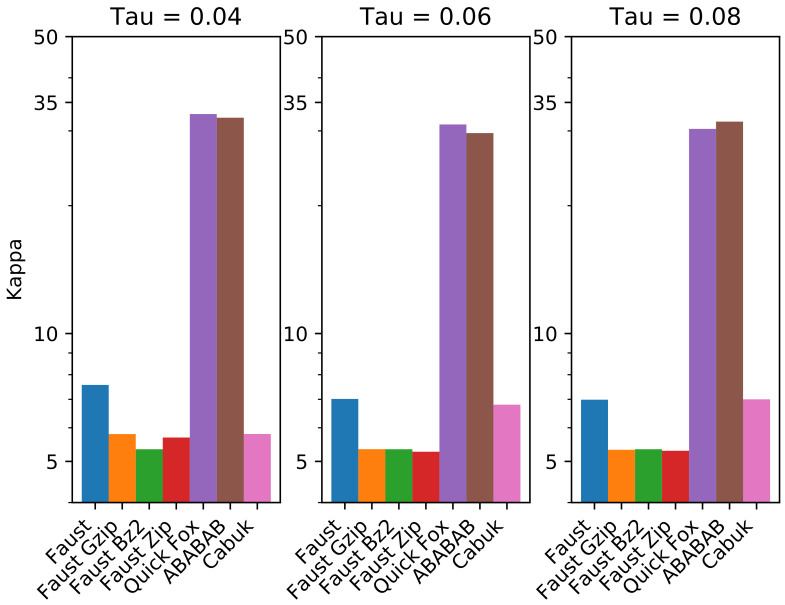

FIGURE 7.4: Dependence of κ on the type of transferred content for SCC, in comparison to results provided by Cabuk et al.

Unfortunately, Cabuk et al. only provided SCC results and no TCC results in their paper. We performed the necessary experiments to show how κ is influenced for timing channels, depending on content type and τ value (Figure 7.5). As shown in the figure, most of the κ values for TCC are in a range of 10-15. Depending on τ, we could also observe a strong difference for the repeating sentence "The quick brown fox ..." (colored in lilac).

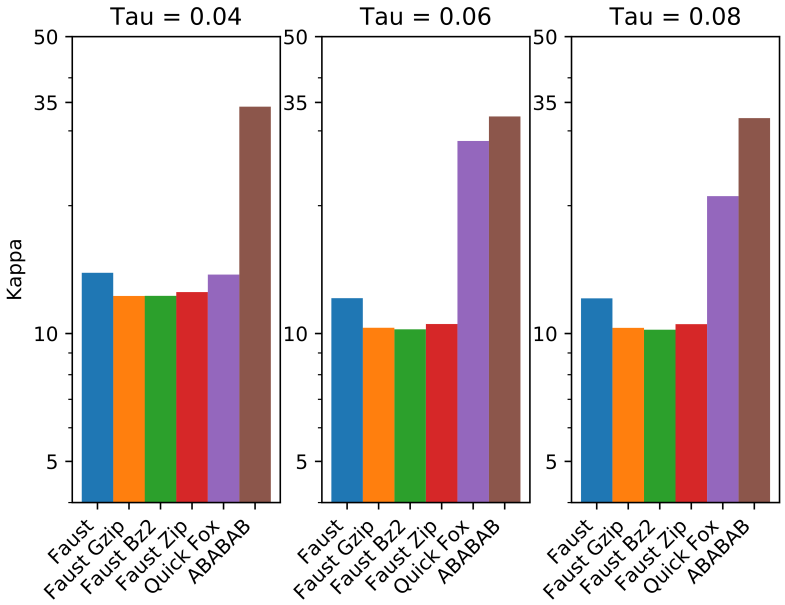

FIGURE 7.5: Dependence of κ on the type of transferred content for TCC.

We also analyzed how the κ values for legitimate traffic are distributed for the NZIX-II data recordings. We applied a precision of two since it was used by Cabuk et al. As shown in Figure 7.6, the largest part of the legitimate traffic (TCP, UDP, ICMP) results in κ values below ten. However, a significant amount of legitimate traffic results in larger values. Given that κ values between 5-7 were presented as SCC traffic by Cabuk et al. and that our experiments showed that most of the TCC traffic has values below 10, a significant amount of the legitimate traffic would result in false positives if a detection would be performed based on the compressibility.

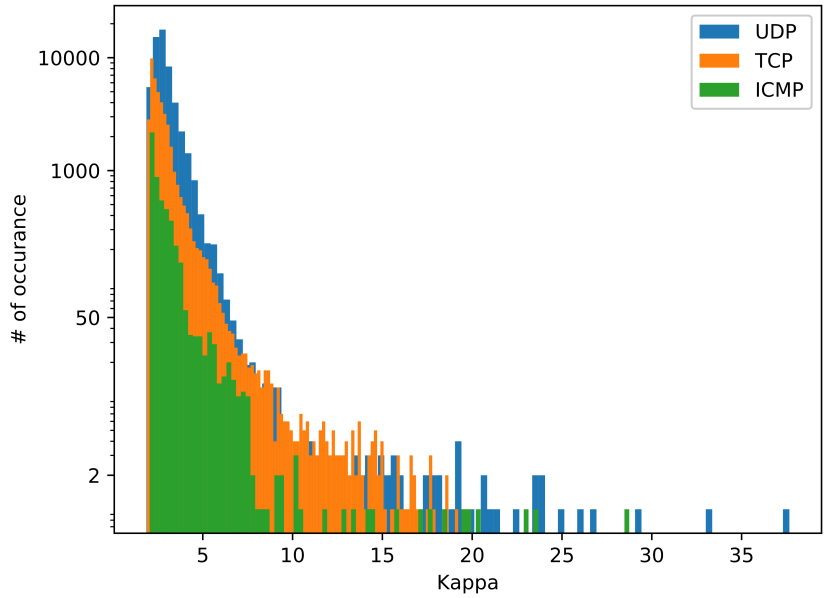

FIGURE 7.6: Histogram of κ values for legitimate traffic from the NZIX-II recordings.

Compressibility-dependence on Network Connection

Finally, we compared, how the type of the network connection influences the compressibility as different connections provide different jitter and performance. We transferred traffic over the local university network in Worms (two hops over ethernet) and between an ISP-hosted server in Frankfurt and our university network (seven hops for most routing paths). Again, we applied the previously used τ values of Cabuk et al. (0.04, 0.06 and 0.08 sec) with a precision of two. We transferred both, TCC and SCC traffic for all combinations of parameters.

As shown in Figure 7.7, remote traffic resulted in overall lower compressibility scores (4-14 instead of 5-68), which we assume is influenced by the increased network jitter. This statement applies for all three τ values as well as for both covert channel types, SCC and TCC.

While Cabuk et al. did not compare different types of network connections, they introduced several degrees of noise (10%-50%) to *legitimate* traffic and

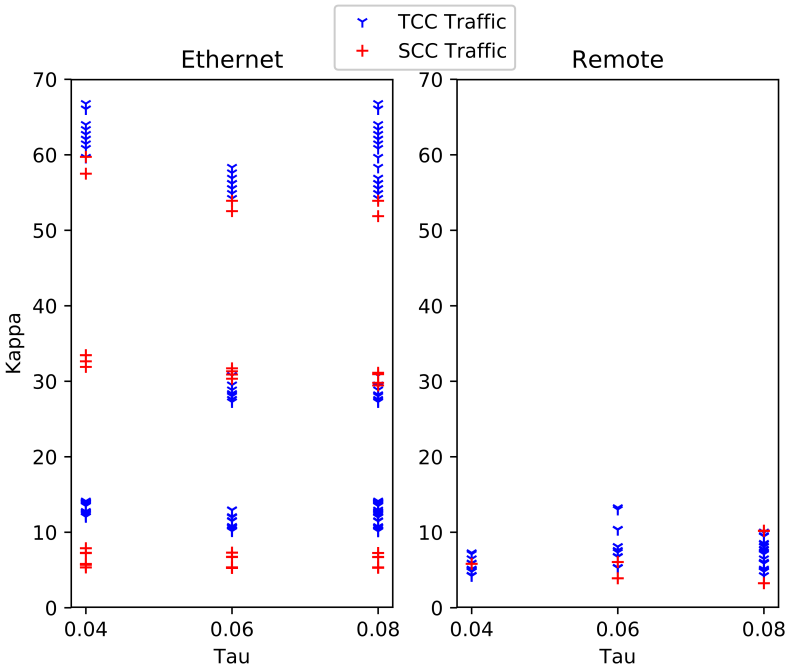

FIGURE 7.7: Dependence of κ on the utilized network connection and τ parameters for both, TCC and SCC traffic.

observed a decrease in compressibility. This is similar to our experiment which compares a seven-hop high-jitter connection with a two-hop low-jitter connection. We thus consider our observation as an indicator that confirms these findings of Cabuk et al.

7.5 Discussion

Despite the discussed advantages, WoDiCoF is linked to a number of drawbacks.

First of all, WoDiCoF currently allows no live-traffic analysis as it requires traffic recordings in the form of PCAP files or NZIX' legacy ethernet format files.

Our testbed is currently flow-oriented, i.e. it does not support detection algorithms for inter-protocol steganography or protocol switching covert channels Wendzel and Keller, 2012a; Jankowski et al., 2013; Mazurczyk et al., 2016a, Chapter 4. Such channels could, however, be detected if each protocol/sub-carrier would be analyzed separately and results would then be merged in an additional step.

What can also be considered a drawback of WoDiCoF is that it requires a fundamental understanding of Apache Hadoop. The requirements are low if researchers or professionals (e.g. LEA users) use the testbed with the already implemented algorithms. However, a deeper understanding of Apache Hadoop is required if new detection algorithms shall be added.

Currently, the visualization is based on a simple heuristic (e.g. threshold values). This means that for a larger traffic recording with thousands of flows, graphs are generated for all flows in combination with used parameters (e.g. for all τ values and all configured precisions of significant digits in inter-arrival times), resulting in several GBytes of graph outputs. The results are challenging to select by hand and thus need decision-making support. We plan to add a visual analytics component for WoDiCoF to aid this problem.

There is also a limitation regarding the evaluation of the compressibility algorithm by Cabuk et al. as we applied neither error-detecting codes nor error-correcting codes. However, this could be added and would most likely not reveal new key insights that are significantly different from the provided content types. Tools such as CCHEF could be used as a part of WoDiCoF to generate such traffic.

Further, Cabuk et al. enhanced their approach by combining different measures for more accurate detectability and also modified their compressibility approach to an approach with sliding windows. We did not performed measurements to compare our results with these extended versions.

Finally, it must be noted that Cabuk et al. discuss several limitations of their work and the problem of finding an optimal window size parameter in their original work. Their compressibility measure can thus be seen as a sophisticated detection approach. However, our results underpin that if certain parameters change (e.g. TCC instead of SCC traffic or change of transferred content), the compressibility scores highly vary. Moreover, we could show that the compressibility values of legitimate traffic can significantly overlap with the compressibility values of covert traffic. Overlapping compressibility values of legitimate and covert traffic would result in false-positives or false-negatives, depending on whether a compressibility threshold for detection is set too low or too high.

7.6 Conclusion and Future Work

WoDiCoF (*Worms Distributed Covert Channel Detection Framework*) is a testbed for experiment verification (due to replication) and parallelization of detection algorithms in network steganography. Our testbed allows the implementation and evaluation of parallel and sequential detection algorithms and the generation of tailored traffic for research.

With WoDiCoF, we were secondly able to provide additional insights in a well-cited covert channel detection algorithm presented by Cabuk et al. The tested algorithm is based on a compressibility score. We performed analyses with several additional parameters not originally tested by the authors and plan to analyze more parameters in future work, especially sliding window sizes. Our results show that we could confirm some of the evaluation results provided by Cabuk et al. but that under varying conditions (e.g. different content type or different type of covert channel (timing vs. storage)), results significantly differ. This underpins the importance of experiment verification in network information hiding.

Thirdly, Cabuk et al. state that more parameters could be analyzed to identify covert channels but that this would *require additional hardware or processing time, and is best done in an offline manner*. This is a key aspect provided by WoDiCoF. We have tested the parallel efficiency of our testbed and could determine a speed-up for the algorithm by Cabuk et al.

Currently, we are working on the implementation of additional detection algorithms for both, covert timing and storage channels. In particular, we plan to implement detection algorithms for more hiding patterns. Moreover are we planning to extend WoDiCoF with a visual analytics component to aid LEA users' analysis phase. WoDiCoF will further be jointly developed together with partners of the CUING initiative and aims to provide a remotely accessible testbed especially for academic and LEA users.

Part IV

Countermeasure Variation & Pattern Extension

Chapter 8

Countermeasure Variation

Abstract Network covert channels enable stealthy communications for malware and data exfiltration. For this reason, developing effective countermeasures for these threats is important for the protection of individuals and organizations. However, due to the high number of known covert channel techniques, it is considered impractical to develop separate countermeasures for *all* theses techniques.

The main contribution of this chapter is to introduce the concept of *countermeasure variation*. Countermeasure variation is a slight modification of a given countermeasure that was designed to detect covert channels of one specific hiding pattern so that the countermeasure can also detect covert channels that are representing *other* hiding patterns.

We exemplify countermeasure variation using the compressibility score and the ϵ-similarity originally presented by Cabuk et al. Both methods were designed to detect covert channels that utilize the Inter-packet Times pattern and we show that countermeasure variation allows the application of these countermeasures to detect covert channels of the Size Modulation, PDU Order and Value Modulation patterns.

Our results show that the detection of all selected types of channels can be accomplished using pattern variation, under several conditions with good precision and accuracy values (in most cases: 95% or higher). However, the detectability differs for the different types of patterns.

Originally published: Wendzel, Eller, and Mazurczyk (2018, *), Wendzel (2019, *), Wendzel, Link, Eller, and Mazurczyk (2019, *), Velino, Mileva, Wendzel, and Mazurczyk (2019, *)

* The content of these publications was extended for this thesis. Several additional results, such as ROC curves, were provided to unify the presentation of the content.

8.1 Introduction

A few hundred hiding techniques for covert channels are known which can be assigned to different families, called *hiding patterns*. Hiding patterns were introduced in 2015 by Wendzel et al. and are abstract descriptions of hiding methods (Wendzel et al., 2015). For instance, the least significant bit (LSB) pattern specifies that secret data can be hidden in the LSB(s) of a header field, but it does not specify where such a field has to be located in a header, which size or byte order the field can have, or to which protocol the hiding technique can be applied to.

So far, countermeasures for covert channels focus only on a single hiding technique or on a family of similar methods which are assigned to the same hiding pattern. To keep the application of countermeasures feasible in practice, their number should be kept at a minimum. Therefore, it must be studied which countermeasures can be applied to which hiding patterns. However, no work is available that has shown that countermeasures can be applied in a way that works with several *patterns*.

In this chapter, we introduce the idea of *countermeasure variation*, i.e., to counter specific covert channels these countermeasures can be potentially applied to multiple patterns instead of only one. As a positive side-effect, countermeasure variation reduces the amount of necessary code per countermeasure as parts of a countermeasure's code can be recycled to counter other patterns. We show the feasibility of countermeasure variation using the compressibility score (Chapter 7.4.1) and the ϵ-similarity used to detect covert timing channels of the Inter-packet Times pattern can also be applied to detect covert channels that modulate packet sizes (Size Modulation pattern).

On one hand this allows academia to generate a flood of papers for every covert channel published, which is not negative per se as it enhances the understanding of countermeasures. Several of the developed countermeasures work well and can even be applied in practice. On the other hand is the increasing number of publications difficult to process by scientists working in the domain. Moreover, such channel-specific countermeasures are limited and their focus cannot be broadened easily. For these reasons, and as discussed in Chapter 3.4, the concept of countermeasure variation can be expected to support the scientific efficiency of this research topic.

The remainder of this Chapter is structured as follows. In this section, the concept of countermeasure variation is introduced. Next, we introduce the traditional detection approaches that we utilize to demonstrate the feasibility of countermeasure variation (Section 8.2). Fundamental detection metrics are afterwards covered in Section 8.3. Next, we demonstrate countermeasure variation for the Size Modulation pattern (Section 8.4), the PDU Order pattern (Section 8.5) and the Value Modulation pattern (Section 8.6). We briefly highlight results of additional countermeasure variations in Section 8.7 and conclude in Section 8.8.

8.1.1 Concept of Countermeasure Variation

This section puts countermeasures in a pattern-based context. It introduces the idea of *countermeasure variation*.

We introduced the concept of *pattern variation* in Chapter 4. The idea of *pattern variation* is that one pattern can change its context, i.e., the network protocol to which it is applied. For instance, the LSB pattern, which hides data in the least significant bit(s) of a protocol header field, can be applied to the TTL field of IPv4 as well as to the Hop Limit field of IPv6. Therefore, the same algorithm can be applied, but the context (network protocol) is changed. Pattern variation is based on the idea of *pattern transformation*. Pattern transformation is an abstract term and it is used, for example, for the dynamic generation of user interfaces so that they fit a given context, e.g., a desktop browser or a mobile phone.

Instead of patterns, countermeasures can also be 'transformed'; we call this process countermeasure variation (the fundamental idea was briefly introduced already in Chapter 4.6). Countermeasure variation modifies a countermeasure to work with another pattern as originally intended.

	Inter-arrival Times	**PDU Order**	**Size Modulation**	**Value Modulation**	**Artificial Re-transmission**
compressibility score	(Cabuk et al., 2009, *)	(Wendzel, 2019)	(Wendzel et al., 2018; Wendzel et al., 2019)	(Velino et al., 2019)	(Zillien and Wendzel, 2018)
ϵ-similarity	(Cabuk et al., 2004; Cabuk et al., 2009, *)	-	(Wendzel et al., 2019)	-	(Zillien and Wendzel, 2018)
regularity score	(Cabuk et al., 2004; Cabuk et al., 2009, *)	-	(Wendzel et al., 2019)	-	-

TABLE 8.1: Summary of existing work on countermeasure variation. (*) indicates the original approaches, i.e., without countermeasure variation.

Please note that although Table 8.1 mentions only three countermeasures, additional countermeasures could be considered, e.g., all countermeasures of Mazurczyk et al., 2016a, Chapter 8. Two of these countermeasures are analyzed in this thesis, namely the compressibility score and the ϵ-similarity.[1]

When a new type of network hiding pattern is found, no countermeasure is instantly available for the new covert channels of the particular pattern. Countermeasure variation allows to transform existing countermeasures so that they can be applied to another pattern. Similarly, countermeasure variation can be applied to already known covert channel patterns for which no or only few countermeasures are known.

In other words, countermeasure variation *transforms*, i.e. modifies, existing countermeasures designed to combat a specific covert channel so that the countermeasure can be used against covert channels which belong to another covert channel that can belong to an entirely different hiding pattern.

[1] In post-thesis work, we have recently demonstrated that the so-called *regularity* metric was suitable for the detection of the Size Modulation pattern (Wendzel et al., 2019).

To perform countermeasure variation for a given countermeasure, both, the input and the inner functioning of an existing countermeasure must be adopted to a new hiding pattern's requirements. For instance, instead of IAT values, packet sizes could be used to detect covert channels of the Size Modulation pattern instead of those of the 'inter-packet times' pattern. The inner functioning must be modified if the existing functioning does either not provide satisfying results with the new inputs or when the new input type is incompatible with existing function (e.g., because a countermeasure is designed to deal with small numbers < 1 but has to deal with large numbers now). There is no generalization feasible of how such a countermeasure variation can be performed as detection methods are highly heterogeneous. However, as we will show in the remainder of this section, performing countermeasure variation is not necessarily a complicated task.

8.1.2 A Definition of Countermeasure Variation

When a new type of network hiding pattern is found, no countermeasure is instantly available for the new covert channels of the particular pattern. Countermeasure variation allows to transform existing countermeasures so that they can be applied to such a new pattern. Similarly, countermeasure variation can be applied to already known covert channel patterns for which no or only few countermeasures are known. However, there is currently no clear definition of countermeasure variation. For this reason, we provide the following definition:

Definition. Given the two hiding patterns A and B, with $A \neq B$, a *countermeasure variation* is a pattern-based process in which an existing countermeasure that detects, limits, prevents or audits covert channels of pattern A is modified so that it detects, limits, prevents or audits covert channels of pattern B.
The process of countermeasure variation replaces the input attributes (*features*) used for A with features for B and performs a modification of the inner functioning (e.g., the algorithm) used for A in order to work with the new features for B. The alternation of the inner functioning is kept as small as possible, which provides the contrast to developing entirely new countermeasures. In comparison to simply applying the same countermeasure (e.g. a statistical method) to another covert channel technique, countermeasure variation *i) requires* the modification of the inner functioning and *ii)* focuses on hiding patterns, i.e., it needs to consider features that can be used for multiple covert channels belonging to the same pattern. ■

In other words, to perform a variation for a given countermeasure, both, the input and the inner functioning of an existing countermeasure must be adapted to a new hiding pattern's requirements. For instance, instead of packets' inter-arrival time (IAT) values used to detect the *Inter-packet Times* pattern, packet sizes could be used as a feature to detect covert channels of the Size Modulation pattern. Or, as shown in (Zillien and Wendzel, 2018), observations of TCP re-transmissions can be extracted from flows to detect the Artificial Re-transmissions pattern. Indeed, multiple features could be combined.

The inner functioning of a countermeasure must be (slightly) modified since the existing functioning (in almost all cases) will not provide satisfying results with the new inputs. Another reason to modify the inner functioning is given when the new input type is incompatible with the existing function (e.g., because a countermeasure is designed to deal with small floating point values < 1 but now has to deal with 32 bit integers as it was the case in (Zillien and Wendzel, 2018) or because sufficient detection results require a modified string generator (Wendzel, 2019)). There is no generalization feasible of how such a countermeasure variation can be performed as countermeasures are highly heterogeneous. However, as we will show in the remainder of this section, performing countermeasure variation is not necessarily a complicated task, which renders the idea a useful and quick method for creating new countermeasures.

As a positive side-effect, recycling the code of one countermeasure to work with a different pattern allows to reduce the overall lines of code: only on a detailed level, the algorithm is slightly altered to fit into the context of the new pattern (i.e., it is *transformed* to the new pattern). However, we do not state that countermeasure variation is *necessarily* less time-consuming than developing new countermeasures from scratch. Instead, its major benefit is to take advantage of existing countermeasures, i.e., it transfers existing countermeasure concepts into new countermeasures.

In this chapter, we show the feasibility of countermeasure variation with three countermeasures originally designed for the detection of the Inter-packet Times pattern. After the process of countermeasure variation, the three countermeasures can be applied to the Size Modulation pattern.

While one could argue that the Size Modulation and Inter-packet Times patterns are rather similar in their functioning (both basically modulate integer values) this was not the case for the Artificial Re-transmissions pattern. Thus, we conclude that countermeasure variation can be expected feasible for other patterns than the already evaluated ones. While the detection results for new patterns after countermeasure variation were acceptable in most cases, there were also cases where no acceptable detection results could be achieved.[2] However, no generic conclusion on the quality of detection results after performing a countermeasure variation is feasible due to the diversity of existing hiding patterns.

8.2 Selected Traditional Detection Approaches

In order to demonstrate the feasibility of countermeasure variation, we will consider two metrics, the compressibility score and the ϵ-similarity. Both metrics were designed to detect IAT-based covert channels. We selected these techniques due to the fact of being highly cited and evaluated. We will perform a countermeasure variation for the Size Modulation, PDU Order and Value Modulation patterns.

[2]The compressibility score did not provide sufficient results for the Artificial Re-transmissions pattern while the ϵ-similarity metric did provide acceptable results for the same pattern.

Cabuk et al. developed a detection approach for covert channels that transfer secret data via delay between network packets (IAT values) in (Cabuk et al., 2009; Cabuk et al., 2004). These covert channels fall under the Inter-packet Times pattern. The basic functionality of such covert channels is that before sending new packets they encode secret data using different IATs[3]. For instance, if the time between two packets is $100\ ms$, this could indicate a binary zero while a time-gap of $200\ ms$ could indicate a binary one. However, detecting such channels is challenging since their coding can vary and because they can easily blend with the legitimate traffic. The three proposed detection metrics for IAT-based channels of Cabuk et al. are the compressibility and the ϵ-similarity.

The *compressibility*-based approach by Cabuk et al. works as follows. For each traffic flow, all n IATs are recorded in a list $\Delta_{t_1}, ..., \Delta_{t_n}$ (we use t to indicate that we focus on timing events). All values $> 1\ s$ are filtered out. All remaining values are coded in ASCII characters in the form that the number of leading zeros behind the comma is encoded in upper-case characters starting from A (no zeros) over B (one zero behind the comma) and so forth. All resulting strings are then concatenated to a large string S (e.g. "`A25B2A25B19A24B22`"). Then, S is compressed with a compressor $\mathfrak{I}$, resulting in the compressed string $C = \mathfrak{I}(S)$. As a compressor, Cabuk et al. used *Gzip*. The compressor is a key component and it is integrated to reveal the decrease of the entropy due to the covert channel utilization as its few IATs occur many times. Finally, the authors divide the length of both strings by calculating the value $\kappa = |S|/|C|$. In result, certain ranges of κ values are an indicator for the presence of a covert channel.

In case of the *ϵ-similarity*, all IATs of a flow are first sorted in a list with ascending order. For every packet P_i in the sorted list, the pair-wise timing difference λ_i with packet P_{i+1} is calculated, i.e., $\lambda_i = |P_i - P_{i+1}|/P_i$ (if the timestamps are equal, the value 0 is used). Next, a value ϵ is selected as a threshold and all relative increases $\lambda_i < \epsilon$ are counted. The number of λ values below the threshold in comparison to all values is then used as an indicator for the presence of a covert channel.

8.3 Utilized Evaluation Metrics

The two metrics used to evaluate all detection methods are precision and accuracy. Precision is defined as the number of true positives (TP) divided by the number of all positives (true and false positives) and it is expressed as:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}.$$

In other words, precision illustrates the percentage of the flows detected as covert channels that were actually covert channels (while other flows

[3] A detailed description of the pattern can be found in (Wendzel et al., 2015; Mazurczyk et al., 2016a).

may have been detected as "covert channels" but were actually legitimate traffic).

Accuracy, on the other hand, expresses how large the number of correctly classified elements is in comparison to *all* elements. In other words, it represents the percentage of flows that were correctly classified as covert or legitimate in comparison to all classified flows (the total population of true and false positives and negatives). The accuracy is calculated as follows:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

Later, we will also utilize additional metrics: *recall* (also: *sensitivity* or *true-positive rate, TPR*) is the number of correctly classified covert channel flows in comparison to the correctly classified covert channel flows plus those flows that were classified as legitimate traffic but were actually covert channel flows, too (false negatives, FN).

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

The *F-score* is the harmonic mean of precision and recall.

$$\text{F} - \text{score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Finally, the *false-positive rate* (FPR) is the amount of legitimate traffic that was considered to be covert channel in comparison to all legitimate traffic and the *Specificity* (also: *true-negative rate*, TNR) measures the fraction of correctly classified negative samples.

$$\begin{aligned} \text{FPR} &= \frac{\text{FP}}{\text{FP} + \text{TN}}, \\ \text{Specificity} &= \frac{\text{TN}}{\text{TN} + \text{FP}}. \end{aligned}$$

So-called *receiver operating characteristic* (ROC) curves show how a classifier's recall performs depending on the FPR. Several ROC charts will be shown in this chapter. For the contained ROC curves, the so-called *area under curve* (AUC) value will be mentioned. All AUC values in this chapter are approximated using the particular intervals of each section.

A Note on the Extension of Measurements and Traffic Mixtures for this Thesis This chapter is based on several different publications which were originally tailored to show detectability under different circumstances. The following steps where undertaken to unify the content. In comparison to the publications used for *Scenario 1*, additional results were provided in the form of ROC curves, AUC values, an FPR chart, improved interval selection, and addition of F-score and recall to all original charts. For *Scenario 2*, ROC curves, FPR charts and several additional detail results were added

for this thesis. The description of *Scenario 3* was enriched with textual details.

In case of all three scenarios, we applied a traffic mixture containing 50% covert channel traffic and 50% legitimate traffic. This was decided to prevent the null-accuracy problem, i.e., achieving a high accuracy by simply selecting the most frequent class. However, to additionally address a scenario where (almost) no covert channel traffic is present, FPR charts are also shown.

8.4 Scenario 1: Size Modulation Pattern

In this section, we first explain the original detection methods as introduced by Cabuk et al., followed by our approaches for countermeasure variations. To evaluate how the transformed detection approaches perform, we used different data samples as shown in Table 8.2.

In the remainder of this section, it must be noted that for each detection technique, we first analyze the detectability of covert channels that encode data using two different symbols, i.e., two different packet sizes. Afterwards, we analyze the detectability of traffic that combines all these covert channels. Finally, we analyze the detectability of covert channels with more than two symbols.

Data-type	**Payload sizes (transp. layer) [bytes]**	$\lvert\Delta_p\rvert$ **[bytes]**	**# of flows**
legitimate	various	various	100,000
covert channel (2 symbols)	1,000 / 1,001	1	100,000
	100 / 101	1	100,000
	50 / 60	10	100,000
	100 / 200	100	100,000
	100 / 1,000	900	100,000
	all the above	all the above	100,000
covert channel (>2 symb.)	100 / 200 / 300	100, 200	100,000
	100 / 200 / 300 / 400	100, 200, 300	100,000
	100 / 200 / ... / 800	100, 200, ..., 700	100,000

TABLE 8.2: Size modulation traffic used to evaluate our approach

8.4.1 Countermeasure Variation

To perform a countermeasure variation for the *compressibility* metric proposed by Cabuk et al., i.e., transferring the original approach to covert channels which utilize the Size Modulation pattern, we modified the following aspects of the original algorithm. First, we considered the relative differences of packet sizes of a flow instead of its IATs. Thus, for each flow with n packets, we calculated $n-1$ relative size differences Δ_{p_i} (p stands for *packet*

size) between the succeeding packets. Second, we concatenated a string S consisting of the relative differences for each flow, separated by commas: $S = \Delta_{p_1}, \Delta_{p_2}, ..., \Delta_{p_n}$. In this string, numbers were represented in ASCII (i.e., the string coding is different to the letter-coded rounded IATs of Cabuk et al.). For instance, if a flow contains five packets with the packet sizes 120, 520, 514, 518, and 520 bytes, then the relative differences $\Delta_{p_1}, ..., \Delta_{p_4}$ would be 400, -6, 4, and 2 bytes. We concatenated the string S using the ASCII representation of the Δ_p values, i.e., "`400,-6,4,2`". We decided to introduce the comma-based separation of values as otherwise, due to the ASCII representation, numbers would not be distinguishable, e.g., the relative differences used above would result in the string "`400-642`", which would influence the compression in an way that does not consider the actual size differences. The remainder of this detection method functions exactly the same as in the case of the original approach. For each flow, we calculated the compressibility of S using a compressor $\mathfrak{I}$ to calculate $C = \mathfrak{I}(S)$, followed by dividing string lengths $\kappa = |S|/|C|$. As already mentioned, the compressor is a key component and it is integrated to reveal the decrease of the entropy due to the covert channel utilization as few utilized covert channel's packet sizes occur many times. Finally, we determined κ values of legitimate traffic and of covert channel traffic to define interval borders in which flows could be considered as covert channel traffic. A similar step is required to categorize κ values in case of the original approach (Cabuk et al., 2009; Cabuk et al., 2004).

Next, we performed a countermeasure variation for the ϵ-*similarity* as follows. First, we sorted all packet sizes of a flow instead of the IAT values. Then, we calculated the relative differences λ_i based on these values and determined suitable ϵ-thresholds. All other steps of this countermeasure were kept as in the original approach.

8.4.2 Evaluation

We first study the compressibility score, followed by the ϵ-similarity.

Compressibility

To evaluate compressibility, first we performed a *training* phase to determine κ values of legitimate and covert traffic (100,000 legitimate packets and 100,000 covert channel packets). As a data source for legitimate traffic, we used the NZIX data set (WAND group, 2000) from the University of Waikato's WAND group. In particular, we considered traffic containing at least 200 packets. Figure 8.1 shows the results of the initial analysis. As visible, κ values of legitimate and covert channel traffic overlap clearly. When the covert channel utilizes more symbols, the median κ value seems to decrease, rendering these channels potentially easier to detect.

The obtained κ values were then used to define interval values that separate covert from legitimate traffic. As visible in Figure 8.1, covert channel traffic has a κ value of approximately 4 to 6, with an approximate mean of 5.0; the differences are depending on the covert channel's number of symbols

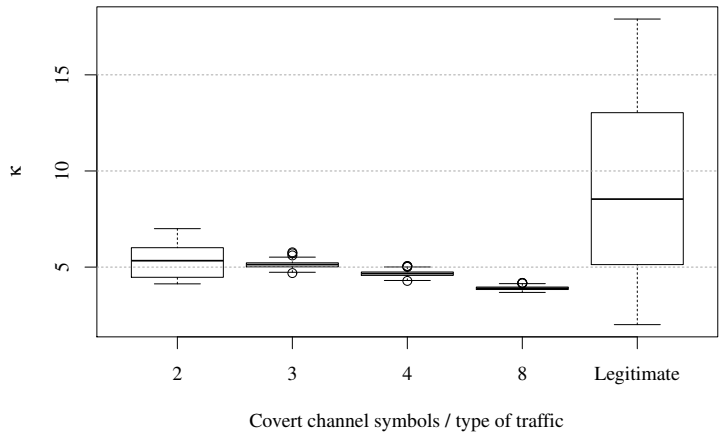

FIGURE 8.1: Analysis of NZIX training data in comparison to covert channels using 2, 3, 4 and 8 symbols.

and our generation of the string S. Therefore, we decided to start with an interval $\langle 4.9; 5.1\rangle$ for κ to differentiate between covert and legitimate traffic. We widened the interval in further steps (Table 8.3 provides an overview of all intervals). Please note that we later introduce improved intervals. Intervals 0 and 13 are just used for plausibility checks.

No.	Range	No.	Range	No.	Range	No.	Range	No.	Range
0	$\langle 0.0; 1.0\rangle$	1	$\langle 4.9; 5.1\rangle$	2	$\langle 4.8; 5.2\rangle$	3	$\langle 4.7; 5.3\rangle$	4	$\langle 4.6; 5.4\rangle$
5	$\langle 4.5; 5.5\rangle$	6	$\langle 4.4; 5.6\rangle$	7	$\langle 4.3; 5.7\rangle$	8	$\langle 4.2; 5.8\rangle$	9	$\langle 4.1; 5.9\rangle$
10	$\langle 4.0; 6.0\rangle$	11	$\langle 3.9; 6.1\rangle$	12	$\langle 3.5; 7.0\rangle$	13	$\langle 0.0; 99.0\rangle$	-	-

TABLE 8.3: Used starting intervals

In the following *testing* phase, we applied another 100,000 legitimate packets and 100,000 covert channel packets to test each interval for every particular type of covert channel (see following sub-sections).[4] Since there are no traffic recordings for the Size Modulation-based covert channels available (Elsadig and Fadlalla, 2017), we decided to generate our own covert channel traffic data with a traffic generator. Our covert channels used different packet sizes for their coding to transfer randomized content, i.e., every hidden symbol (packet size) occurred with the same probability. This is a realistic assumption as secret data can be encrypted before being transmitted. Some of the covert channels used a coding with significantly different packet sizes, e.g., sending either a packet of size 100 bytes or of size 1,000 bytes. Other covert channel's coding was only marginally distinguishable, e.g., sending either a packet of size 1,000 or 1,001 bytes. Table 8.2 provides an overview of the generated covert channels, all following a uniform distribution of covert symbols.

It must be noted that we apply the same detection intervals for the detection of all covert channels, i.e., we do not further optimize the intervals to

[4]In case of the combined test of all covert channels using two symbols, we transferred 20,000 packets per covert channel, so that 100,000 packets were processed overall.

match a specific channel's characteristics to ease detection.[5] Instead, we just decrease the interval size gradually.

Overview (ROC curves and AUC) The detectability of covert channels using different PDU sizes is shown in form of a ROC curve in Figure 8.2. As can be seen, covert channels with the same byte difference (e.g., the 1-byte difference channels using 1,000 and 1,001 or 100 and 101 bytes) is only marginally distinguishable. The AUC of the 1,000 and 1,001 bytes channel was 92.33% while it was 92.36% for the channel using 100 and 101 bytes. This slight difference is probably caused by the PRNG as only 200 flows were generated.

As can also be seen, the detectability does not automatically increase when larger byte differences are used, i.e., the channel using a 900-bytes difference is slightly better detectable (AUC 86.37%) than the one with a 100 bytes difference (AUC 84.79%). The channel with the 10-byte difference resulted in an AUC of 95.14%. The AUC differences are especially the result of our ASCII coding.

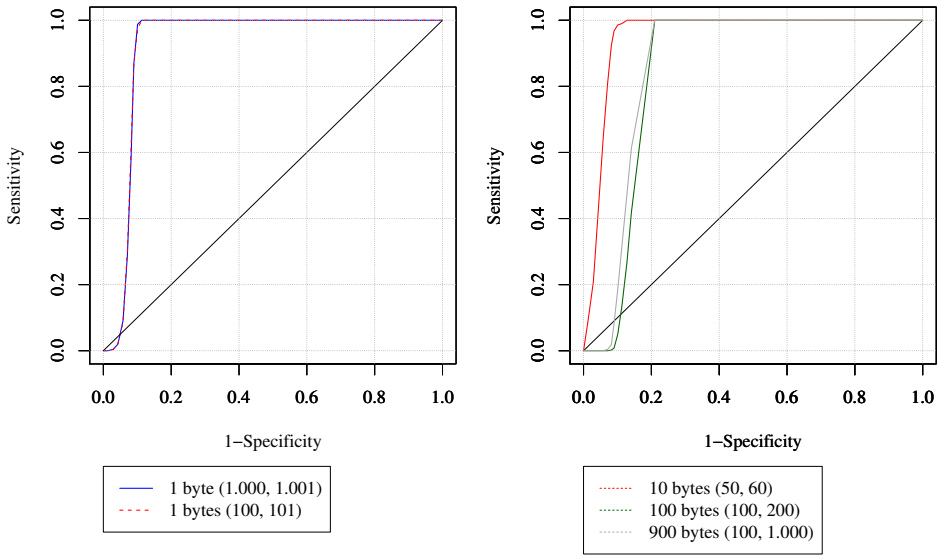

FIGURE 8.2: ROC curves for the detection of size modulation covert channels.

For comparison, when the covert channel utilizes multiple symbols in parallel, the detectability in terms of AUC seems to decrease with the number of symbols (see Figure 8.3). In particular, the 3-symbol channel resulted in an AUC of 97.68%, the 4-symbol channel had an AUC of 95.18%, and the 8-symbol channel's AUC was 84.84%.

Improved detection intervals As the detectability using the intervals of Table 8.3 was good but not of optimal quality, we decided to introduce specific intervals for the particular covert channels, optimized for their median value. In particular, we applied the intervals of Table 8.4 in the remainder.

[5]This was decided to reflect realistic conditions.

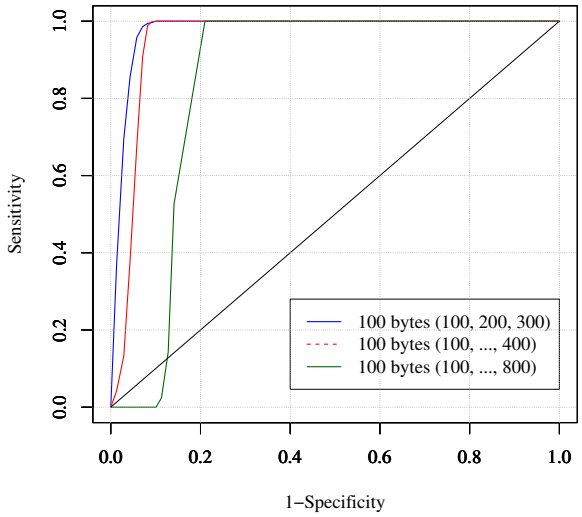

FIGURE 8.3: ROC curves for the detection of size modulation channels with 3, 4 and 8 symbols.

Please note that these intervals are identified by their numbers during the following pages.

No.	Range	No.	Range	No.	Range	No.	Range	No.	Range
0	$\langle 0.0; 0.1\rangle$	1	$\langle 3.7; 4.2\rangle$	2	$\langle 4.1; 4.6\rangle$	3	$\langle 4.2; 4.7\rangle$	4	$\langle 4.3; 4.8\rangle$
5	$\langle 4.6; 5.1\rangle$	6	$\langle 4.7; 5.8\rangle$	7	$\langle 4.9; 6.0\rangle$	8	$\langle 5.3; 7.0\rangle$	9	$\langle 0.0; 99\rangle$

TABLE 8.4: Improved intervals (tailored to match specific covert channels)

Two-symbol Covert Channels We first analyzed covert channels with a payload size difference of only 1 byte, i.e., the covert channel that encodes data with 1,000 and 1,001 payload bytes and the one that encodes secrets with 100 and 101 payload bytes. As the differences in packet size were always -1, 0 or +1 bytes for both covert channels, they resulted in highly similar κ values that only depended on the randomly selected covert channel symbols. The mean κ value for the channel with 1,000 and 1,001 bytes was 4.43448 while it was 4.43677 for the channel with 100 and 101 bytes. Thus, the precision and accuracy values for both channels were almost equal for our interval sizes. As shown in Figure 8.4 (upper-left corner), the best performing interval was #3, resulting in an accuracy of 97.17% and an F-score of 95.82%.

Next, we analyzed the covert channel with a payload size difference of 10 bytes, i.e., payload sizes of 50 and 60 bytes (upper-right corner of Figure 8.4). Here, we could achieve the best accuracy and F-score values for the interval #7, namely 94.86% and 92.82%. The mean κ value (5.34675).

The covert channel with the payload sizes 100 and 200 bytes provided an accuracy of 93.59% and an F-score of 91.21% for the best-performing interval (#8), see lower-left corner of Figure 8.4. The mean value for κ was

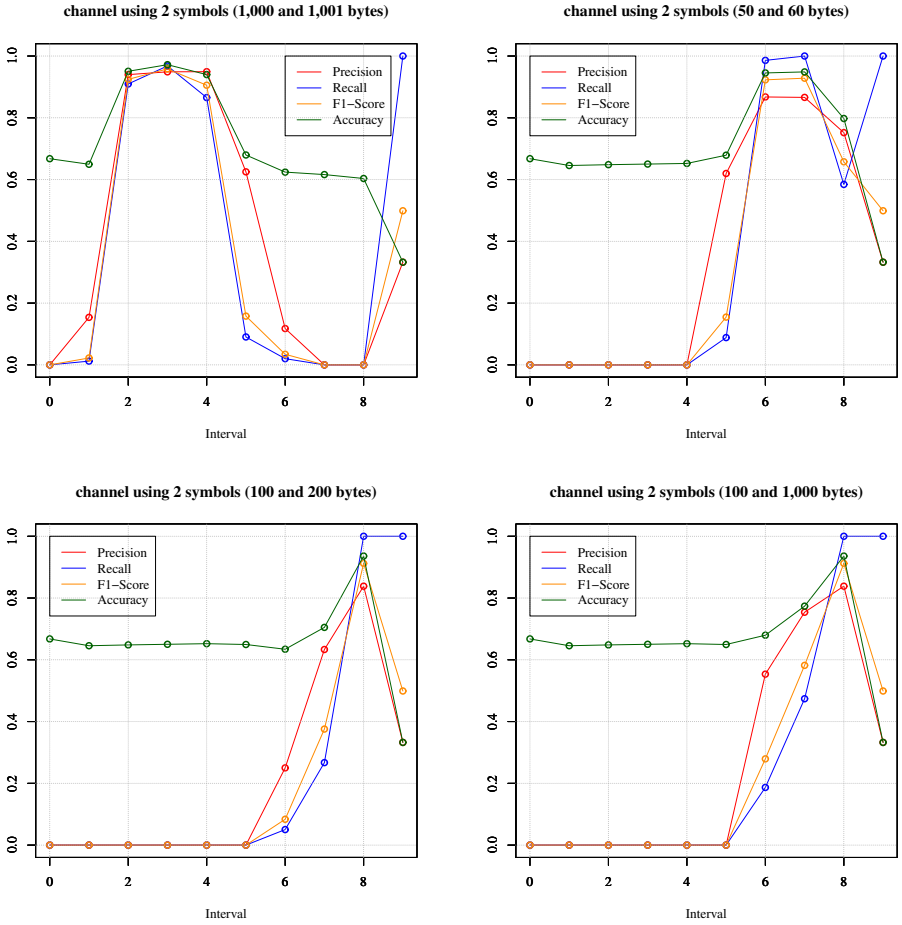

FIGURE 8.4: Detection results for covert channels using the 2 symbols but different payload size differences.

6.16748. The the mean value for the channel with 100 and 1,000 bytes was almost the same (6.03886), resulting in basically the same detection values for interval #8 (see lower-right corner of Figure 8.4).

Table 8.5 summarizes the average κ values that were calculated for the different two-symbol covert channels. As can be seen, the κ value generally increases when the difference of the utilized packet sizes increases.

Payload sizes [bytes]	**Δ of pkt. size**	**Avg. κ value**
1,000 / 1,001	1	4.43448
100 / 101	1	4.43677
50 / 60	10	5.34675
100 / 200	100	6.16748
100 / 1,000	900	6.03886

TABLE 8.5: Resulting κ values for two-symbol covert channels.

FPR per Interval However, it is crucial to consider thresholds that result in a low FPR. As visualized in Figure 8.5, the realistic thresholds #1-#8 were all in a range below 10%, thresholds #1 to #5 were below 3% FPR. However,

none of the realistic intervals was below 2% FPR. Interval #4 ($\langle 4.3; 4.8 \rangle$) was the one with the lowest FPR (2.281%).

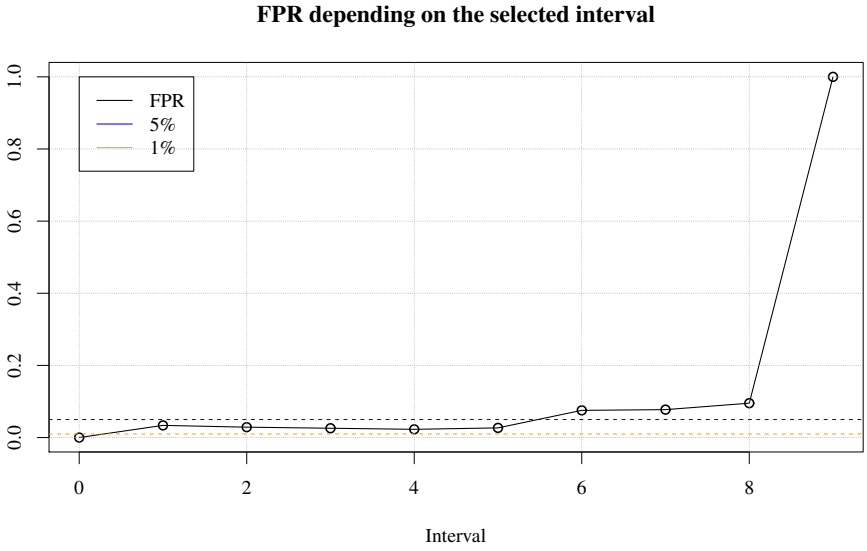

FIGURE 8.5: FPR using 1,000 legitimate flows for all utilized thresholds.

The channels with 1,000 and 1,001 as well as 100 and 101 bytes (latter not visualized) performed best for interval #3, they profits from a low FPR (2.58%) while the optimal detection intervals for the channels with 50 and 60 bytes, 100 and 200 bytes, and 100 and 1,000 bytes where best for intervals 6, 7, and 8 resulting in FPR values of 7.54%, 7.74%, and 9.52%, respectively.

Combining Two-symbol Covert Channels Figure 8.6(l) illustrates the detectability for a mixture of all above-mentioned covert channels and legitimate data mentioned in Table 8.2. The fraction of flows and packets per type of covert channel was equal, resulting in 20,000 packets per covert channel (100,000 covert channel packets overall). Again, the same number of legitimate flows and packets were used in this test to provide balanced classes.

First, we applied the already known intervals from Table 8.4. However, these intervals are tailored to match one covert channel type, each, and not multiple covert channels. As can be seen in the left part of Figure 8.6, the best interval in terms of combined accuracy and F-score was #8 (accuracy: 71.54%, F-score: 64.90%).

Since these results were not satisfying, we created improved intervals to match more than one covert channel. These intervals are shown in Table 8.6 and their results are visualized in Figure 8.6(r). As can be seen, the results improved clearly, with the best performing intervals being #2 and #3. Interval #2 provided an F-score of 92.07% and an accuracy of 91.40% while interval #3 provided an F-score of 91.53% and an accuracy of 90.75%. However, the F-score for intervals #2 and #3 where rather high (17.20% and 18.50%, respectively), which is a result of the widened intervals that lead to the selection of a higher fraction of legitimate flows.

No.	Range	No.	Range	No.	Range	No.	Range	No.	Range
0	$\langle 5.0; 6.8\rangle$	1	$\langle 4.5; 7.0\rangle$	2	$\langle 4.1; 7.0\rangle$	3	$\langle 4.0; 7.1\rangle$	4	$\langle 2.9; 7.2\rangle$

TABLE 8.6: Tailored detection intervals for multiple two-symbol covert channels

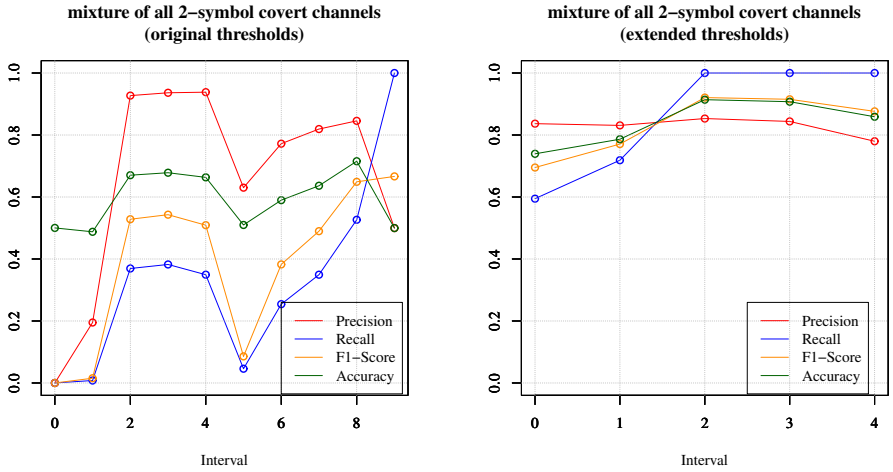

FIGURE 8.6: Detectability for a mixture of *all* presented two-symbol covert channels using the intervals of Table 8.4 (left figure) and the further widened intervals thresholds of Table 8.6 (right figure).

Covert Channels with > 2 Symbols Sophisticated covert channels can utilize more than two secret symbols, so that more information can be transferred per packet, i.e., for n symbols, $\log_2(n)$ bits can be transferred per packet. Thus, the required number of packets can be reduced. To analyze such channels, we generated traffic for covert channels using 3, 4 and 8 different symbols which were reflected in the payload sizes between 100 and 800 bytes (Table 8.2). In the following, we switch back to the previous intervals listed in Table 8.4.

First, we analyzed a covert channel with 3 different payload sizes (100, 200 and 300 bytes). As shown in Figure 8.7 (upper-left corner), interval #6 performed best (F-score: 92.90%, accuracy: 94.93%). Results of interval #7 were similarly good. However, all other intervals provided no useful results.

In case of a covert channel with four different payload sizes (100 to 400 bytes), the best performing interval was #4 (see Figure 8.7, upper-right corner). The F-score was 90.66% and the accuracy 94.06%. Other intervals resulted in F-scores below 80% and where thus considered unsuitable.

Finally, we analyzed a covert channel with 8 different payload sizes (Figure 8.7, bottom-left corner). This channel was detectable with an accuracy of 97.66% and an F-score of 96.59% using interval #1. The FPR was 3.30%. All other intervals provided no useful results.

Combined Detection of Covert Channels with 3+ Symbols In summary, channels with 3, 4, and 8 symbols required different intervals (intervals #6 and #7 vs. #4 vs. #2). Since the median values of these channels were, as already shown in Figure 8.1, within the range of approx. 3.3 to 5.5, it was

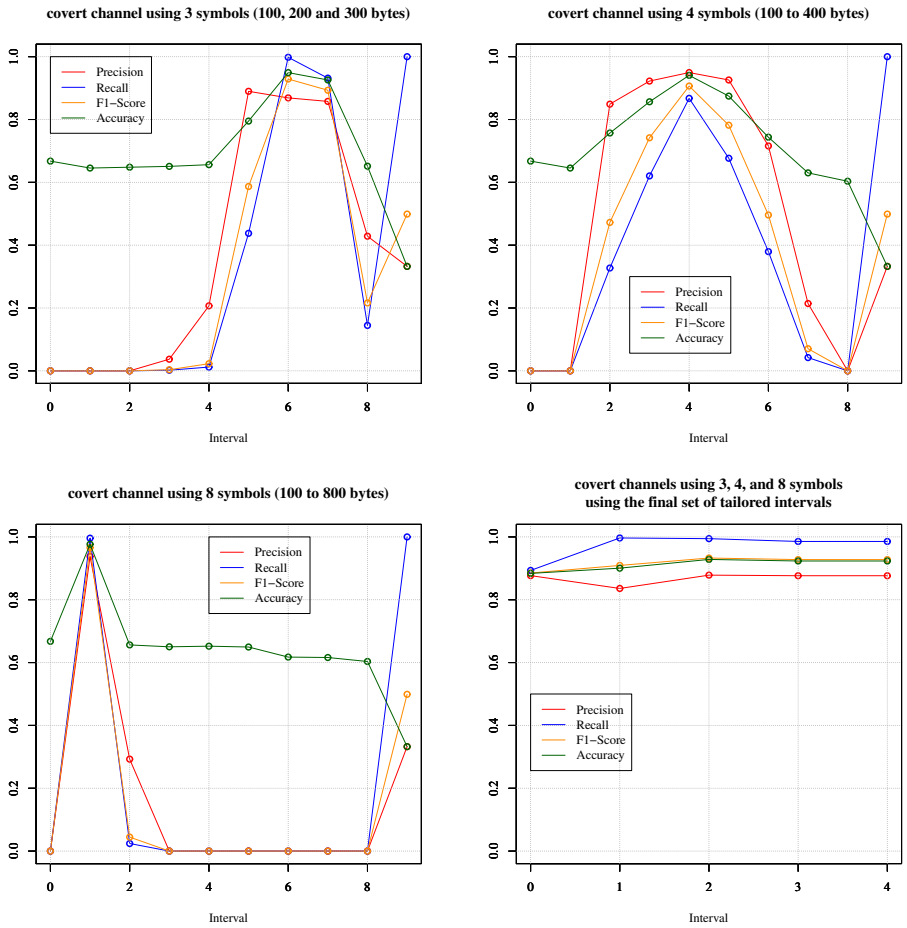

FIGURE 8.7: Detectability of covert channels using 3, 4, and 8 symbols in parallel.

feasible to realize a combined detection with another set of intervals (Table 8.7). The results are visualized in Figure 8.7 (bottom-right corner).

No.	Range	No.	Range	No.	Range	No.	Range	No.	Range
0	$\langle 3.3; 5.2\rangle$	1	$\langle 3.0; 5.6\rangle$	2	$\langle 3.5; 5.5\rangle$	3	$\langle 3.4; 5.4\rangle$	4	$\langle 3.4; 5.4\rangle$

TABLE 8.7: Tailored detection intervals for covert channels with 3+ symbols

Intervals #2-#4 all performed similarly. The best-performing interval in terms of combined F-score and accuracy was #2, which resulted in an F-score of 93.28% and an accuracy of 92.83%, combined with a FPR of 13.78, which is a result of the large interval.

ϵ-similarity

Similarly, like a κ value for the compressibility score was initially determined, we first needed to establish suitable thresholds for the ϵ-similarity. However, for these measurements, we applied 3,000,000 legitimate flows. It turned out that $\epsilon = 0.1$ (i.e., 10%) provided good results when we expect 0.34% or 0.35% of the λ values below ϵ (see Figure 8.8 for a two-symbol channel using 1,000 and 1,001 bytes). The AUC was 97.3%.

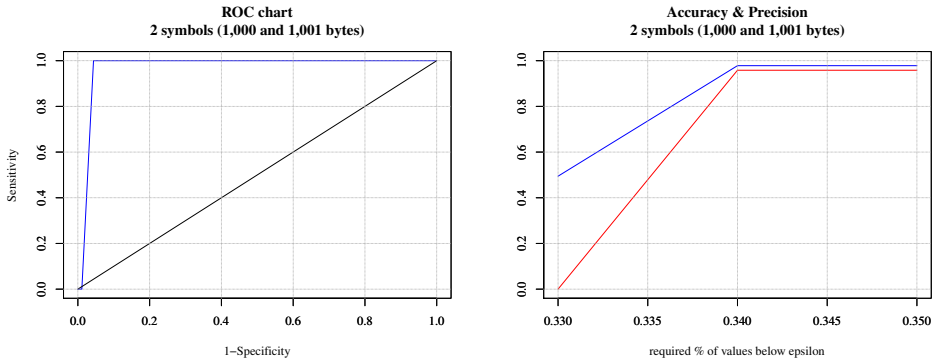

FIGURE 8.8: ϵ-similarity: ROC curve (l) and accuracy/precision (r) for the detection of a covert channel using 1,000 and 1,001 bytes.

All results for two-symbol covert channels were highly similar, which is rooted in the fact that our post-countermeasure variation ϵ-similarity counts the number of relative packet size increases[6] while the allowed increase is reflected by the configured threshold. For this reason, we have visualized only the channel using 1,000 and 1,001 bytes. We were able to achieve an accuracy of 97.8%, a precision of 95.8%, and a recall of 100% for above mentioned limits (F-score: 97.85%).

However, we could not obtain any useful results for covert channels with three or more symbols when applying the ϵ-similarity, rendering the result of our countermeasure variation unsuitable for such channels. The reason for this is that the number of relative increases in ϵ values was already overlapping with too many legitimate flows.

8.5 Scenario 2: PDU Order Pattern

In this section, we present an approach to detect PDU order channels in a protocol-independent manner. Our approach is tailored for the scientific community and for the law-enforcement to analyze traffic recordings. However, our approach can be potentially applied to corporate networks with a low traffic volume in real-time, too. In particular, the contributions are as follows:

1. Perform a modification (*countermeasure variation*) of the so-called *compressibility* score that was originally introduced to detect covert channels that modify the timing between PDUs. Therefore, a suitable coding is presented. The presented detection approach is designed in a way to make it independent of the utilized network protocol.
2. Evaluating the detectability of PDU order channels that are utilizing a different number of PDUs in their coding.

[6]The number of relative packet size increases directly depends on the number of symbols.

8.5.1 Fundamentals

The functioning of PDU order channels is visualized in Figure 8.9. A covert sender (CS) sends a sequence of messages to the covert receiver (CR). In comparison to legitimate traffic, the order of protocol data units (PDUs), i.e. network packets, is modified by a covert sender to encode a hidden message. Sender and receiver agree upon the coding a priori, e.g. it can be defined in a malware's executable before deployment.

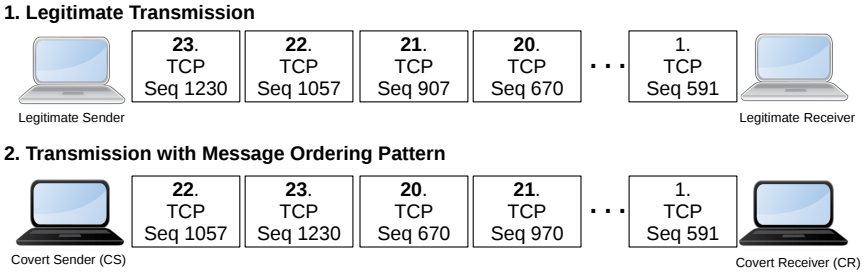

FIGURE 8.9: The functioning of the PDU order pattern.

In general, a covert channel manipulating the order of n packets can be used to transfer $\log_2 n!$ bits (Ahsan and Kundur, 2002b; Kundur and Ahsan, 2003). Thus, the more PDUs are utilized, the more bits can be transferred per PDU. For practicability, longer messages can also be split into m sequences of n PDUs each, allowing $m \cdot \log_2 n!$ bits to be transferred. For instance, if a transfer contains 2,000 packets, and if $n = 4$, then this would allow the transfer of $500 \cdot \log_2 4! = 2,292.5$ bits (more than 286 bytes), and it would be 2,762.8 bits (345 bytes) for $n = 5$ ($m = 400$). PDU order channels can utilize the sequence numbers of TCP, AH or ESP headers, several application layer protocol's sequence numbers and the IPv4 Identifier field, among others (Ahsan and Kundur, 2002b; Kundur and Ahsan, 2003; Mazurczyk et al., 2016a).

Hiding patterns are abstract descriptions of covert channel hiding techniques; they were introduced in (Wendzel et al., 2015). Each hiding pattern represents a set of covert channel techniques that follow the same general hiding technique. These hiding patterns form a taxonomy, where the PDU order pattern is categorized as a timing channel (because manipulating the order of PDUs effectively alters their timings). In (Mazurczyk et al., 2016a), the PDU order pattern was additionally categorized as *protocol-aware*, i.e. as a method *that require[s] the understanding of the carrier protocol*[7], and was finally renamed to *(manipulated) message ordering* pattern.

Covert channel detection is also well covered in the existing literature. In general, countermeasures either aim on detecting the presence of the channel itself or the involvement of a participant in the covert communication. Other countermeasures aim on limiting the channel capacity or on preventing the use/the existence of channels. Moreover, channels can be audited (Mazurczyk et al., 2016a). Several detection methods exist and summaries can be found in (Zander et al., 2007a; Mazurczyk et al., 2016a). A recent

[7] In contrast, protocol-agnostic timing channels can modify the carrier blindly, e.g. by manipulating the inter-arrival times between packets.

trend is the detection of distributed covert channels (Cabaj et al., 2018b). PDU order channels could also be realized in a distributed manner, e.g. split over multiple flows simultaneously. We do not explicitly focus on such a distributed scenario in this chapter, but the presented approach could potentially detect distributed PDU order channels nevertheless on a per-flow basis. Moreover, digital media steganography has similar channels, e.g. methods that modify the order of HTML tags in websites, which can be detected with statistical methods (Polak and Kotulski, 2010). However, no specific method was presented so far to detect network-based PDU order channels.

A detection approach for covert timing channels it the so-called *compressibility score* that was introduced by Cabuk et al. (Cabuk et al., 2009). The compressibility score is applied to detect channels that modulate inter-arrival times (IAT) between succeeding network packets. These channels belong to the "Inter-packet Times" pattern (Mazurczyk et al., 2016a). Therefore, IAT values of a flow are recorded, and then rounded. The rounded values are encoded as short strings that are concatenated to a long string S. Afterwards, S is compressed with a compressor $\mathfrak{I}$ (e.g. *Gzip*), i.e. $C = \mathfrak{I}(S)$. Finally, the value $\kappa = |C|/|S|$ is calculated. κ is used as an indicator for the presence of a covert timing channel. In general, a covert channel's artificial IAT values of the "inter-packet times" pattern would be similar within a given flow, rendering the results better compressible than legitimate traffic with more varying IAT values (Cabuk et al., 2009). In comparison to the inter-packet times pattern, the PDU order pattern does not encode hidden information in the IAT values but in the pure order of PDUs, i.e. it does not matter how large the IAT values between PDUs are.

8.5.2 Countermeasure Variation

The development of countermeasures for criminal uses of covert channels and steganography has recently attracted increasing attention, cf. (Cabaj et al., 2018a). Therefore, we consider a scenario where traffic recordings of criminal cases must be analyzed some time after they were recorded. This means that we do not focus on a real-time detection scenario, i.e. computing performance of our approach is not a strong requirement. However, it can potentially be applied in low-volume networks, too, for real-time detection.

Our detection approach is designed to work for intermediate nodes (IM) located between covert sender (CS) and covert receiver (CR) that conduct traffic recordings. It can be a gateway or router. However, as an administrative system process, our detection approach could also be integrated into CS or CR as long as the network covert channel process and its user are separated from the detection process. This could be realized, e.g. in the form of a malware. However, in this case, the detection process needs direct access to the network interface, e.g. as a kernel-level routine. As it is beneficial to recognize *all* packets of a flow, it would even be optimal to locate the detection process on CS or CR since there might be multiple IMs for multiple routing paths, so one IM might not see *all* traffic exchanged between CS and CR.

We perform a countermeasure variation as follows. First, we modify the compressibility score of Cabuk et al. in a way that we provide it with a different input as for the algorithm. Second, we modify the generation process of the string S. Third, we determine optimal thresholds to differentiate between legitimate and covert channel traffic. The other steps of the compressibility score (compression and calculation of κ) remain as in the original approach.

Algorithm Input and String Coding

In the original approach by Cabuk et al., only one algorithm is shown to encode IAT values into strings. Instead, we encode sequence numbers and, moreover, experimented with four different approaches to encode these sequence numbers. Another difference to the original approach by Cabuk et al. is that IAT values cannot overrun but sequence numbers can indeed overrun. For this reason, we implemented a filter. We also defined new detection thresholds for κ.

The details of our approach are described in the following. Figure 8.10 visualizes our approach.

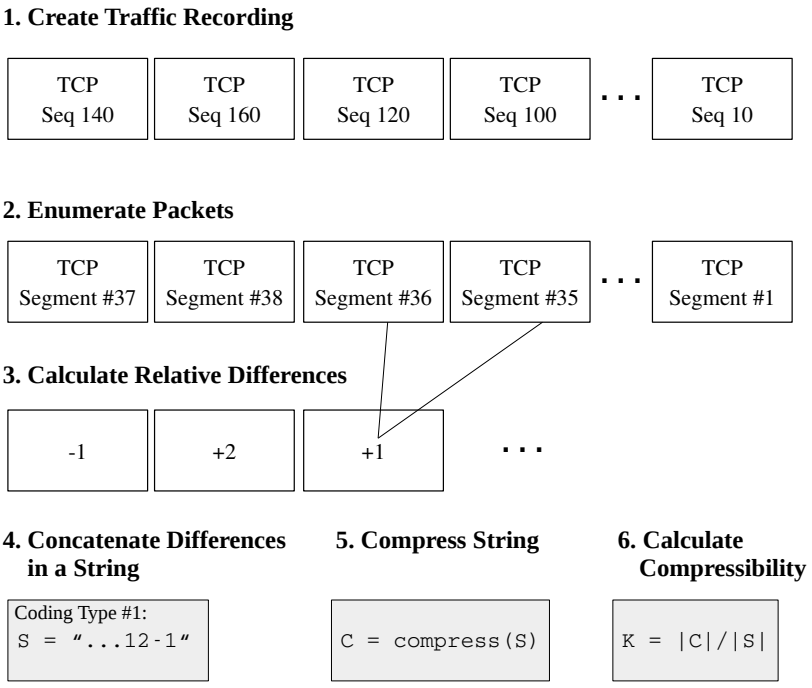

FIGURE 8.10: Our proposed five-step approach to detect PDU order channels (steps 3 and 4 are exemplified for one type of coding).

In step 1, we record all flows between two or more hosts to be observed (Figure 8.10-1). For each flow, we consider a window of 200 PDU sequences, i.e. 201 packets. For instance, such a flow could contain 201 TCP segments (or alternatively a flow using any other protocol with a header field that allows to determine the PDU order).[8] For all these sequences, we assign

[8]Please note that the PDU order pattern requires protocols to have a numbering element. Thus, protocols without such a field are not of relevance for our research.

each PDU a number between 1 and 200 based on its appearance in the order of packets in the window. For instance, if four following sequence numbers were 100, 120, 160, 140, for the packets 35 to 38, then we would record the order 35, 36, 38, 37 (see Figure 8.10-2).

In the following, `diff` represents the relative difference between the current and the last PDU number. For the mentioned example, the values for `diff` would be +1, +2, and -1. `|x|` indicates the absolute value of the variable `x`. Moreover, `a || b` is the string concatenation of the strings `a` and `b`. We use `a || (y ? 't' : 'f')` to test whether the condition `y` is true. If it is true, the string `'t'` will be concatenated to the string `a`, otherwise the string `'f'` will be concatenated to `a`. The function `str(x)` returns the value of `x` in the form of a string.

We tested the following four different codings. Please note that only one string element is shown each, i.e., 200 of these values would be concatenated to create S.

1. `str(diff)`: concatenate the `diff` values of succeeding packets (exemplified in Figure 8.10).
2. `str(|diff|)`: concatenate the absolute `diff` values of succeeding packets.
3. `str(diff) || ( |diff| % 2 ? 'A' : 'B' )`: for all succeeding packets, concatenate `diff` with an 'A' if the absolute value of `diff` mod 2 equals 1, otherwise with 'B'.
4. `str(|diff|) || ( |diff| % 2 ? 'A' : 'B' )`: for all succeeding packets, concatenate the absolute value of `diff` with an 'A' if the absolute value of `diff` mod 2 equals 1, otherwise with 'B'.

Figure 8.10-3 and 4 exemplify coding type 1, i.e. a simple concatenation of the `diff` values. However, in result, coding type 4 performed best using our training data-set. Using coding 4, a perfect transmission without retransmissions and with no out-of-order packets would look like "`1A1A`... `1A1A`" with $|S| = 400$, resulting in $|C| = 27$ and thus $\kappa = 14.81481$ (two characters per difference). Indeed, we observed that several legitimate flows (approx. 28%) had exactly this κ value and the related string generate from 200 sequence numbers each.

Sometimes, sequence numbers overrun. For instance, if a sequence number field has eight bits, the sequence number following 255 would be 0. These overruns are rare, but we filtered them by checking whether the difference in the current PDU number is larger than 5 times the previous PDU number difference (`olddiff`) using the expression `olddiff > diff*5`. This filter slightly improved detection results additionally. Moreover, this filter can handle typical sequence number fields. We tested the algorithm with TCP (32 bit sequence numbers) and the CCEAP covert channel[9] tool (8 bit sequence numbers).

Finally, we compressed the concatenated string S using $C = \mathfrak{I}(S)$ and then calculated $\kappa = |C|/|S|$ as done in the original approach by Cabuk et al. (Figure 8.10-5 to 8.10-6).

[9] `https://github.com/cdpxe/CCEAP/`

Initial Parameterization

For each a flow, in order to decide whether a covert channel or a legitimate data flow is present, a threshold for the value κ must be defined. Therefore, we extracted 454 flows with at least 201 sequence numbers from the NZIX data set[10] and calculated their compressibility scores. Please note that the NZIX data are coming from a real Internet environment and contain PDUs which are reordered due to legitimate reasons. We compared the NZIX scores with the compressibility scores of three different covert channel types, modifying the order of 2 to 4 PDUs, both randomly to represent the transmission of encrypted content. As shown in Figure 8.11, all types of covert channels resulted in comparably κ values: channels with 2 PDUs had a mean of 3.983 while those with 3 PDUs had a mean of 2.621 and those with 4 PDUs had a mean of 2.259. The NZIX training data resulted in a mean κ value of 10.674. However, as visible in Figure 8.11, there is a clear overlapping of compressibility results between especially NZIX training data and covert channels using 2 PDUs. For this reason, we decided to parametrize our detection approach with thresholds in the range of $\kappa = \langle 2; 14 \rangle$[11], while the range $\langle 2.5; 5.5 \rangle$ was studied in detail since only few covert channel compressibility scores were above 5.5.

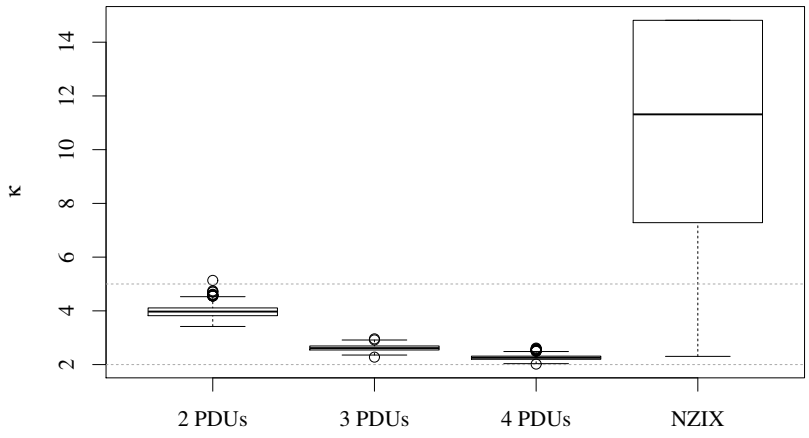

FIGURE 8.11: Analysis of NZIX training data in comparison to covert channels using sequences of 2 and 4 PDUs (horizontal lines at κ = 2 and κ = 5).

[10] `ftp://wits.cs.waikato.ac.nz/pma/long/nzix/2/`

[11] The maximum achievable compression using `Gzip` was slightly above 14.8, thus rendering higher values insignificant. For compression, we write the string S into a file and run `gzip -9 -no-name <filename>` to calculate C.

Encoding using how many PDU pairs	Number of CC and legitimate flows	Overall CC seq. numbers	Overall legitimate seq. numbers
2 PDUs	720 each	144,000	144,000
3 PDUs	720 each	144,000	144,000
4 PDUs	720 each	144,000	144,000

TABLE 8.8: PDU order traffic used to evaluate our approach

8.5.3 Evaluation: Compressibility

We used the aforementioned tool CCEAP to create PDU order covert channels.[12] Therefore, we extended the tool so that randomized sequence numbers that represent encrypted content could be created.

We generated flows containing pairs of 2, 3 and 4 PDUs each and recorded the generated traffic. Each flow configuration was repeated 20 times and with different inter-arrival times between 10ms and 500ms. Overall, we created 2,160 covert channel flows (720 for every 2, 3 and 4 PDU channel). We extracted the sequence numbers as described in Section 8.5.2 and calculated the compressibility scores for all these flows.

Afterwards, we extracted flows with at least 200 sequence numbers from NZIX traffic recordings with approx. 8,500,000 packets, resulting in 1,100 flows, of which the first 720 were considered. Next, we compared the detectability of every CCEAP configuration with the NZIX recordings. Each time, the same number of legitimate and covert channel flows were used to calculate the detectability, i.e., we applied a scenario where 50% of the flows were used for the covert communication and 50% of the flows represent legitimate communication (Table 8.8). Overall, this resulted in 144,000 pairs each of the three covert channel variants. However, we later study a scenario with 100% legitimate traffic to analyze the FPR (Section 8.5.3).

For all thresholds (2, 2.5, 2.75, 3, 3.25, 3.5, 3.75, 3.9, 4, 4.025, 4.05, 4.075, 4.1, 4.15, 4.2, 4.25, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5, 5.5, 6, 7, 8, 9, 10, 11, and 12) we measure the detectability for PDU order channels utilizing 2, 3 and 4 PDUs. The measurement of the detectability is based on the metrics precision, recall, accuracy and F-score and all these measures are visualized in the following figures.

Experimental Results

Using the aforementioned parameter set, we achieved a high detectability of all three covert channels. Figure 8.12 shows the ROC curve for these

[12] We extracted the CCEAP application layer sequence number for this purpose. However, one could also use raw TCP segments generated by tools such as `scapy` instead. Due to the protocol-independent detection design and the above-mentioned filter, the particular tool and protocol makes no difference in terms of experimental evaluation as long as the order of the PDUs is determined by a CSPRNG since only the order of segments, not their actual 32-bit or 8-bit value, is considered.

channels, where the AUC values of Table 8.9 were achieved. In the following, we will discuss the results in detail.

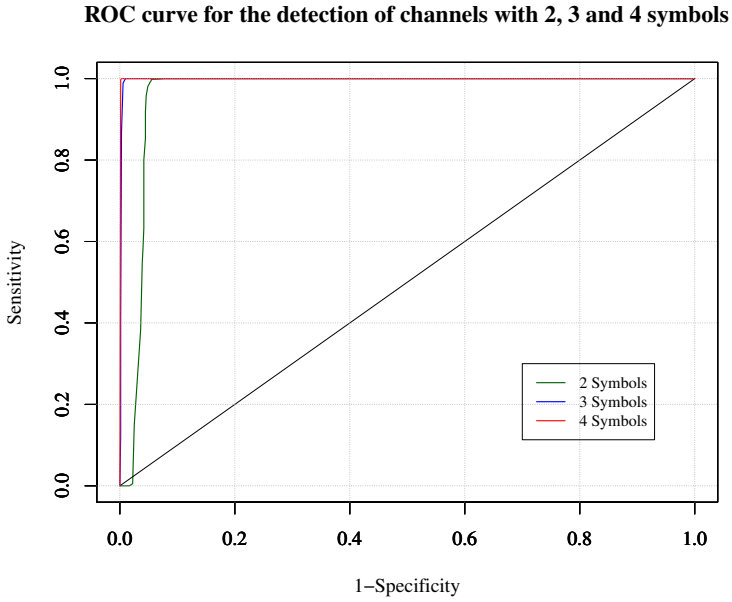

FIGURE 8.12: ROC curves for the three analyzed covert channels.

Seq. Items	AUC
2	0.963698
3	0.997974
4	0.999298

TABLE 8.9: AUC values for the ROC curves of Figure 8.12.

First, we tested all parameters for covert channels using two symbols. The results are shown in Figure 8.13. Here, the threshold of $\kappa = 4.6$ performed best (also see Table 8.10 for detailed results). However, the achieved accuracy of 94.513% and F-score of 94.756% are not satisfying for large volumes of traffic. The thresholds $\kappa < 4.4$ and $\kappa \geq 4.9$ were not leading to acceptable results (accuracy and F-score below 94%).

As shown in Figure 8.14, the optimal threshold for channels that were utilizing 3 PDUs appears to be better for lower values of κ than in case of channels utilizing 2 PDUs. For the 3 PDU channels, acceptable values (both, accuracy and F-score ≥94%) were achieved for $\kappa = 3$ to $\kappa = 4.8$ with the best results being achieved for $\kappa = 3$ (≥99.23% accuracy and F-score).

For a more sophisticated channel that would utilize 4 symbols, the optimal threshold of $\kappa = 2.75$ was again lower (Figure 8.15 and Table 8.10). Results for $\kappa \geq 3$ were equal to channels utilizing 3 PDUs as the maximum compressibility scores of both channels were each below the same maximum value. Acceptable results of ≥94% accuracy and F-score were achieved for $2.5 \leq \kappa \leq 4.8$.

Seq. Items	Accuracy	F-score
κ = 2.5:		
2	49.722%	0%
3	55.833%	21.673%
4	99.027%	99.022%
κ = 2.75:		
2	49.513%	0%
3	92.847%	92.375%
4	**99.513%**	**99.516%**
κ = 3:		
2	49.236%	0%
3	**99.236%**	**99.241%**
4	99.236%	99.241%
κ = 3.25:		
2	48.958%	0%
3 & 4	98.958%	98.968%
κ = 4.25:		
2	90.555%	90.38%
3 & 4	96.18%	96.32%
κ = 4.3:		
2	91.805%	91.815%
3 & 4	95.833%	95.999%
κ = 4.4:		
2	93.333%	93.494%
3 & 4	95.416%	95.617%
κ = 4.5:		
2	94.166%	94.384%
3 & 4	95.138%	95.364%
κ = 4.6:		
2	**94.513%**	**94.756%**
3 & 4	94.93%	95.174%
C4.5 Classifier:		
2	95.9% (+1.39%)	95.9% (+1.14%)
3	99.54% (+0.30%)	99.55% (+0.31%)
4	99.84% (+0.33%)	99.8% (+0.28%)

TABLE 8.10: Detection results, depending on the number of sequence items for selected thresholds (improvement of C4.5 classifier over best manually selected threshold).

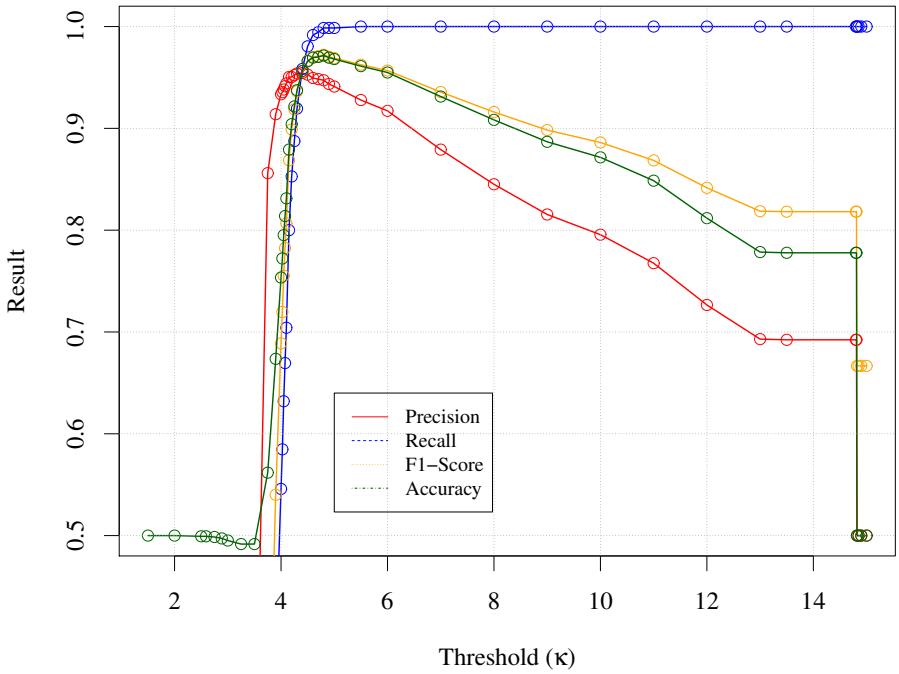

FIGURE 8.13: Detection results for 2-PDU channels.

Finally, we investigated how well a mixture of covert channels utilizing 2, 3 or 4 PDUs can be detected. Therefore, we combined the legitimate NZIX flows with covert channel flows utilizing 2, 3 and 4 PDUs. We used 50% NZIX flows and 50% covert channel flows, of which 1/3 were using the same number of PDUs each. Figure 8.16 shows that the optimal threshold appears to be approximately around $\kappa = 4.5$. However, differences between $\kappa = 4.4$ to $\kappa = 4.7$ were only marginal (Table 8.11), indicating that the threshold is value is not highly sensitive.

Consideration of Realistic Conditions

Under realistic conditions, we can expect the percentage of covert channel traffic using the PDU order pattern to unknown to an observer (warden). Potentially 100% of the flows between two nodes represent covert traffic, but it could also be 0%. In the previous section, we studied scenarios with 50% covert and 50% legitimate traffic. However, a reasonable scenario that we can also consider is where the share of covert channel traffic is very low. If we assume that the amount of covert channel traffic is close 0%, then an important factor to be considered is the *false-positive rate* (FPR). If the FPR is high, the number of false-positives could easily lead to extensive analysis work in an operational network.

We thus analyzed the FPR of legitimate-only traffic for different thresholds. Therefore, we analyzed 1,156 NZIX flows with ≥ 200 sequence numbers extracted from one split PCAP file with approximately 8,750,000 packets.

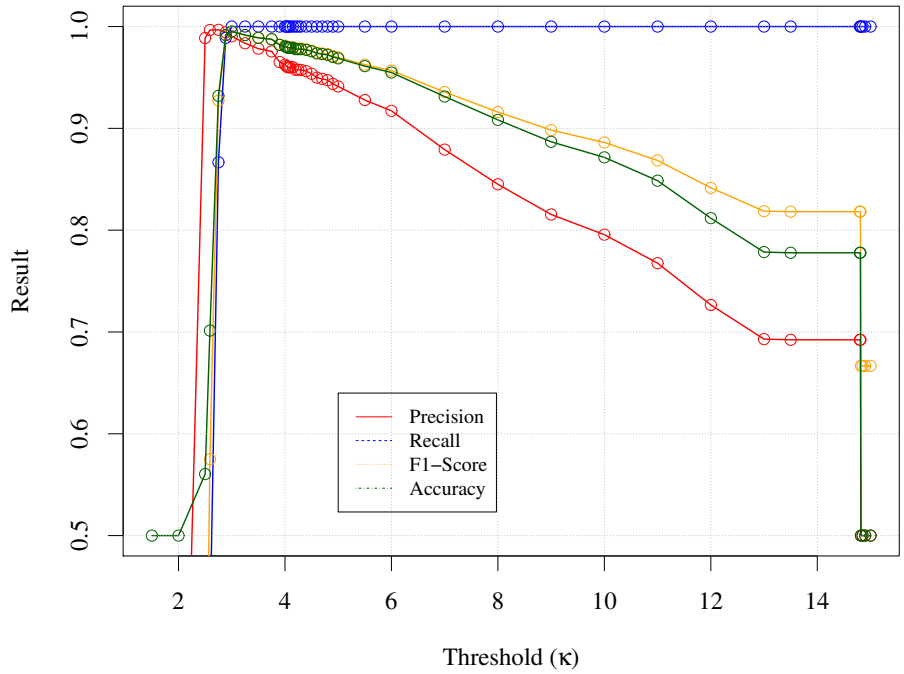

FIGURE 8.14: Detection results for 3-PDU channels.

The results are shown in Figure 8.17. The FPR increases to more than 5% for thresholds $\kappa > 4.5$, rendering our approach impractical for these thresholds. However, for the thresholds 2 to 3.9 the FPR is between 0% and 3.719%.

If the optimal thresholds of $\kappa = 2.75$ to $\kappa = 3.0$ for channels using 3 or 4 PDUs are selected, the FPR is kept at 0.259% and 1.038%.

However, to detect channels that utilize only 2 PDUs, we optimally apply a threshold of $\kappa = 4.6$. This threshold leads to a FPR of 5.19% and thus would be impractical to handle in real-world scenarios as discussed in (Steinebach et al., 2018). The overall optimal threshold for a *mixture* of covert channel types ($k = 4.5$) results in a FPR of 4.93%.

We thus conclude that our detection approach can be only considered practical under the following conditions:

- The detection of channels utilizing 2 or more PDUs is practical for very low traffic volumes such as in small enterprises or office networks due to the rather high FPR of 4.93%.
- The detection of channels utilizing 3 or more PDUs is practical in real-world scenarios in combination with a low-threshold due to a FPR of 0.259% and 1.038%. We expect that channels utilizing ≥ 5 PDUs are also providing low compressibility scores and should thus be detectable even easier with our approach.

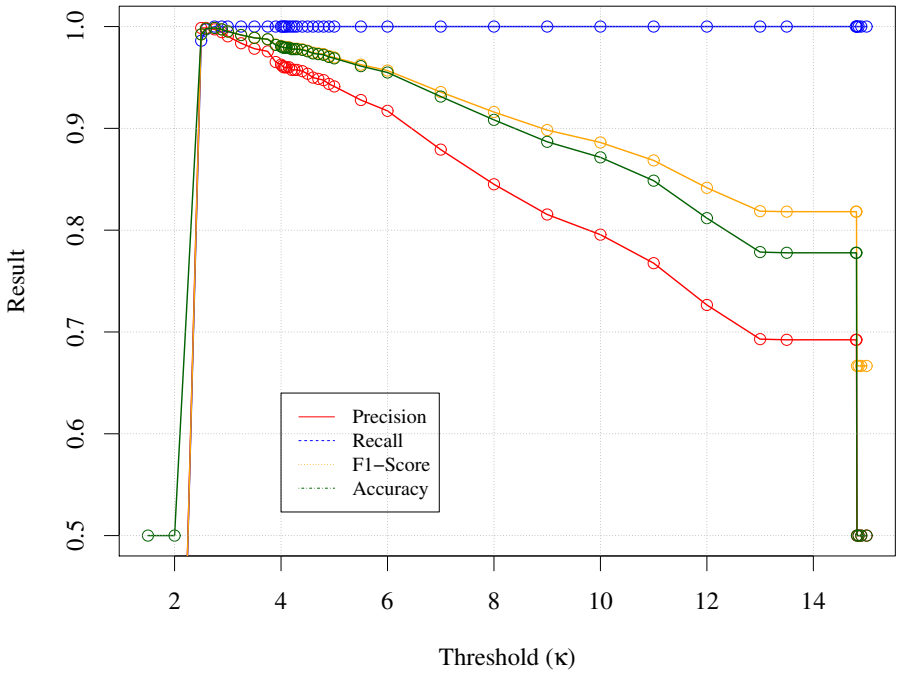

FIGURE 8.15: Detection results for 4-PDU channels.

In other words, channel types with the largest capacity can be detected the best while channels with the lowest capacity (utilizing only 2 PDUs) can be considered difficult to detect with this approach.

Further Optimization Using C4.5 Classifier We trained the J48 classifier, which is an implementation of the C4.5 decision tree algorithm, using the tool `weka` with only one feature, i.e. the compressibility score of flows. We used only this single feature as no other traffic feature would directly be influenced by the PDU order pattern, potentially despite the inter-arrival time of packets. However, even with only one feature, C4.5 can be used for two purposes:

1. To compare our selected threshold's performance with one or multiple thresholds automatically selected by C4.5. In this sense, we applied C4.5 as a means of quality control for our threshold-based detection approach. Our threshold-based approach provided good detection results but if our threshold was selected close to the optimum value, then its result should be only slightly lower than those of the C4.5 classifier that uses the same feature.[13]

[13]This is rooted in the fact that the classifier cannot only automatically determine a suitable threshold but can also divide the selection of legitimate and covert channel flows by applying several thresholds in a row.

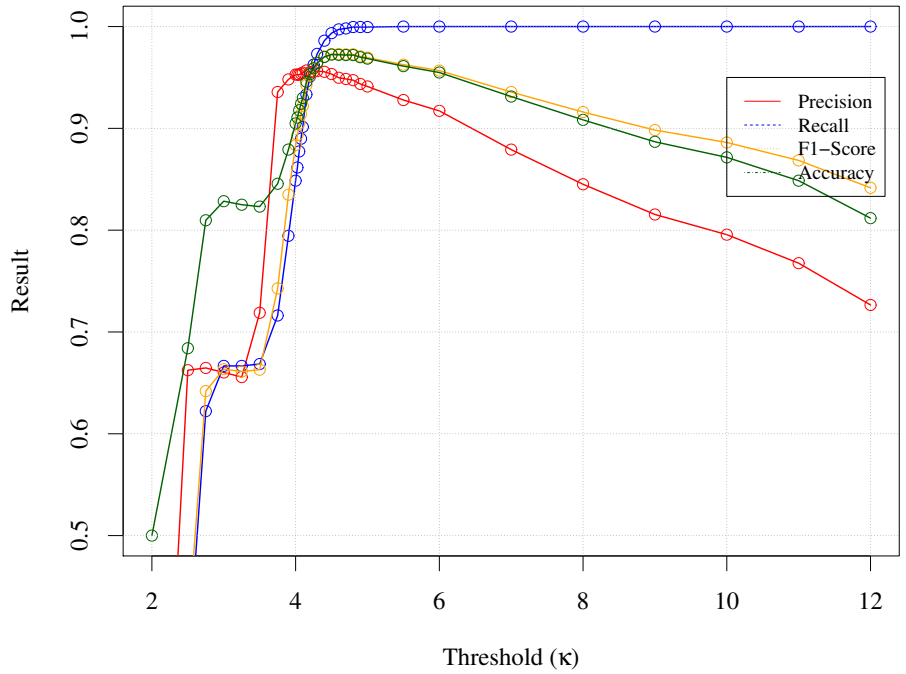

FIGURE 8.16: Detection results for a mixture of the three previously used covert channel types.

2. To let C4.5 find suitable thresholds that we can later use for our heuristic instead of our previously selected and discussed thresholds. Applying these new thresholds will further allow us to calculate the FPR.

First, we studied all three covert channel types separately. With 660 NZIX and 660 covert channel flows (each, i.e. per number of utilized PDUs) and a 10-fold cross-validation, we achieved a slightly higher detectability than using the previous thresholds. C4.5 resulted in slightly different thresholds for κ and most decision trees had only one decision element. For a channel with 3 symbols, the C4.5-selected κ-threshold was 2.88659 (previously 3.00), which resulted in a FPR of 0.605% (instead of the previous 1.038%). For a channel with four PDUs, the threshold for κ was 2.59047 (previously 2.75) and the linked FPR was even below 0.1% (0.086% instead of the previous 0.259%). This means that both thresholds were close to the original thresholds but improved the FPR values further.

The results increased even more for covert channels using two symbols (more than one percent improvement in accuracy and F-score). However, in comparison to the other two channel types, 2-PDU channels resulted in a larger decision tree and our results came from an non-pruned tree (overfitted) and would decrease to the previous level after pruning.[14]

[14]The major threshold selected by C4.5 for the 2-PDU channel was 4.6 (i.e., exactly the one that we already selected).

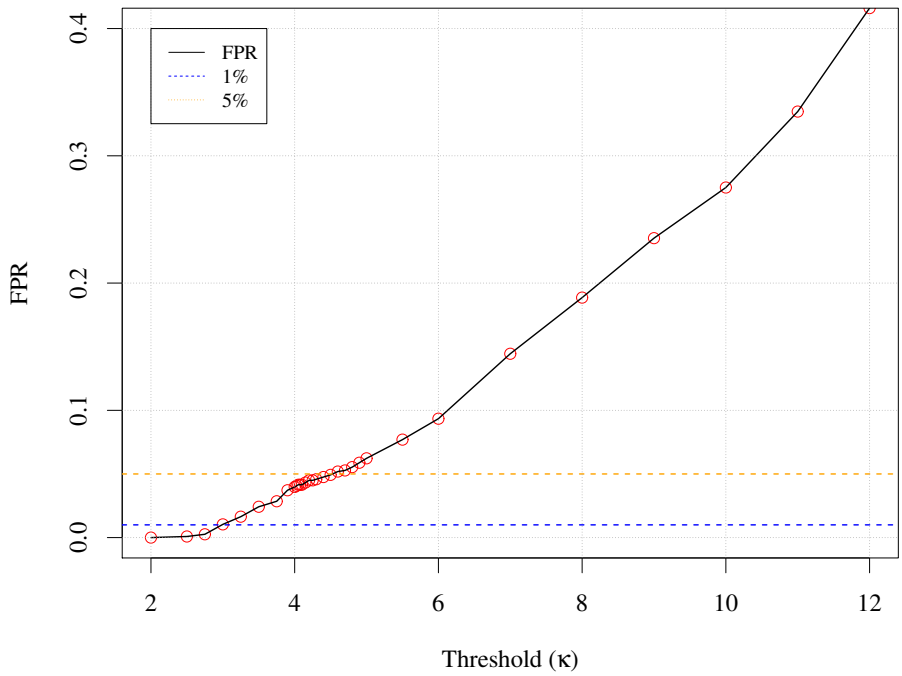

FIGURE 8.17: False-positive rate for our threshold-based detection.

Method	Accuracy	F-score
Threshold, $\kappa = 4.4$	94.72%	94.91%
Threshold, $\kappa = 4.5$	**94.81%**	**95.04%**
Threshold, $\kappa = 4.6$	94.79%	95.04%
C4.5 Classifier	96.72% (+1.91%)	96.7% (+1.66%)

TABLE 8.11: Detection results over all covert channels, depending on the threshold.

Figure 8.18 visualizes the C4.5-based detectability for all 3 covert channel types depending on the number of utilized PDUs. For exact comparison, the results are also shown at the end of Table 8.10. Overall, the new thresholds provided an improvement of approx. 0.3% for accuracy and F-score of channels utilizing 3 or 4 PDUs.

Finally, we compared the overall detectability of the threshold-based approach with the C4.5 classifier for all three covert channel types. As shown in Table 8.11, the C4.5 classifier outperformed our simpler threshold-based detection approach that used $\kappa = 4.5$ (improvement of 1.91% in accuracy and 1.66% in F-score). Again, we applied a non-pruned tree here to determine a maximum value. We can conclude that there is still slight room for improvement when it comes to the detection of a mixture of the three covert channels. However, even the detection results of C4.5 with an over-fitted tree were comparable to our simpler threshold-based detection approach.

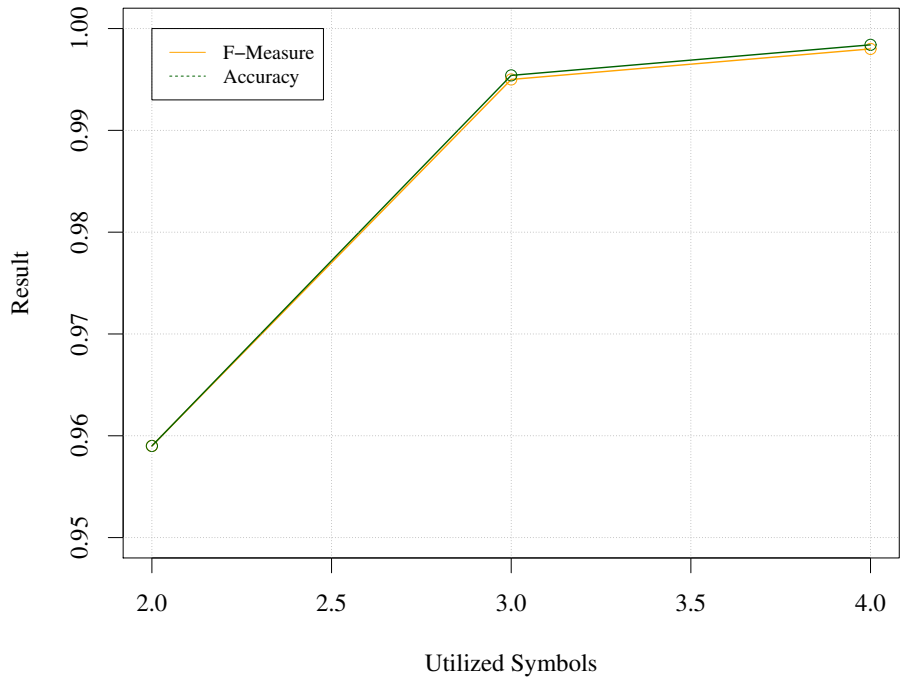

FIGURE 8.18: F-score and Accuracy for D4.5, depending on the number of utilized symbols.

8.5.4 Further Remarks

Our detection approach can either be used on traffic recordings or in a real-time scenario. However, in a *real-time detection scenario*, our approach requires caching of potentially large flows since it uses a stateful algorithm (cf. Mazurczyk et al., 2016a, Chapter 8; Handley et al., 2001). This renders a real-time detection for huge volumes of data difficult: an attacker could target the detection system by creating a high number of flows that must be cached by the warden, resulting in resource exhaustion, potentially leading to a DoS. However, if limits (e.g. on the number of cached flows/unit of time) are enforced, such attacks could be prevented.

On the other hand, if *traffic recordings* are provided and if they are not required to be processed in real-time, our detection approach can be easily used. This fact renders our approach suitable for forensic analyses, e.g. under the umbrella of the law-enforcement.

However, a key criterion that we observed during our experiments is the provision of enough RAM if large PCAP files must be processed. Our implementation (non-optimized shell scripts using basically a command pipeline with `traceconvert`, `tshark`, `grep`, `cut`, `sed` and `awk`[15]) was capable of handling PCAP files of few hundred MByte on an older SUN Fire X2100 server with 4 GByte of RAM, equipped with an AMD dual core

[15]Given enough RAM, an easy way of utilizing multiple cores would be to replace `grep` with `xargs`. Our implementation used only one CPU core.

Opteron 180 CPU (running Ubuntu 18.04 LTS) as long as the traffic recordings could be processed entirely in RAM. However, if large traffic recordings must be processed, then all flows and all sequence numbers of each particular flow must be enumerated, which is a computing-intense problem, consuming several days of computing time on our server.[16] Some NZIX recordings had a compressed size of 2 GByte and we needed to split these files into smaller pieces to process the files on our system (processing these files on a virtual machine with 8 GByte RAM was possible without splitting the files). The performance of our implementation was improved when we loaded all traffic data into memory (*ramdisk*) instead of relying on disk I/O.

In general, our heuristic provides good detection results for some of the studied channels. However, a classifier using neural networks or similar concepts and additional features for the classifier could potentially lead to better results but also to more opaque ones (Daniel E. Geer, 2019), which can be considered a drawback, especially when explainable, court-proof evidence must be provided by LEA users. For this reason, we can consider the application of our simple algorithm useful for LEAs when the following aspects are taken into account: channels using 2 PDUs were linked to a FPR that is not useful for criminal investigations (5.19%). Even the small FPRs of 0.6% and 0.08% for 3 and 4 PDU channels could provide false evidence for non-criminals due to the chance for false-positives. However, we can assume that it is rather unlikely that only one PDU order channel flow would be exchanged over a longer period of several days between CS and CR. For this reason, larger traffic volumes should result in several 'positives'. However, the lower the compressibility score of a 'positive' flow, the more likely a covert channel is present. Thus, for criminal investigations, we recommend not only to determine the number of 'positives' but also their compressibility values. Moreover, the *more* positives, the higher the chance that CS and CR did in fact exchange covert information using PDU order.

8.6 Scenario 3: Value Modulation Pattern

Next, we analyze a covert channel that uses the Value Modulation pattern. The *Message Queuing Telemetry Transport* (MQTT) protocol is a popular IoT protocol and several organizations are using IoT-devices that operate on the basis of this protocol. MQTT-capable devices can connect to an MQTT server and with its help exchange messages. Additionally, for the sake of automation, MQTT setups contain a smart hub, which orchestrates all these devices, provides logic and usually a dashboard, through which the user can control all devices, locally or remotely (e.g., via a mobile phone).

[16] A more efficient sorting could most likely be achieved if *Apache Hadoop* (or similar distributed solutions) would be applied instead. However, performance was not a concern in our experimental evaluation.

8.6.1 Fundamentals

MQTT is a lightweight, client-server, publish-subscribe message transport protocol, suitable for machine-to-machine (M2M)/IoT connectivity. It is designed for resource-constrained devices and low-bandwidth, high-latency and/or unreliable networks, but it is also suitable for mobile applications. MQTT has been heavily applied since its appearance in 1999, e.g. in Facebook Messenger[17], Amazon IoT (a part of the Amazon Web Services)[18], OpenStack[19], home automation platform Home Assistant[20], Microsoft Azure IoT Hub[21], and in the FloodNet project[22] for monitoring river levels and environmental information to provide early warning of flooding, etc.

As already mentioned, MQTT is a publish/subscribe protocol. Basically, this is a client/server model that allows the client to communicate with an endpoint. In this protocol two types of clients have been defined. The clients that send messages are called *publishers*. Other clients that receive the messages are called *subscribers*. These two types of clients never communicate directly with each other. In order to exchange messages, they utilize a central point that plays the role of a server and is called a *broker*. The role of the broker is to receive the messages sent by the publishers and to forward them to the subscribers. Publishers send messages on certain *topics* identified by their topic names. Subscribers, on the other hand, use topic filters to subscribe/unsubscribe to/from specific topics by sending SUBSCRIBE/UNSUBSCRIBE commands. When a broker receives the message, it determines the topic to which it needs to be sent and then transmits the message to those clients (subscribers) who have subscribed to that particular topic (Figure 8.19).

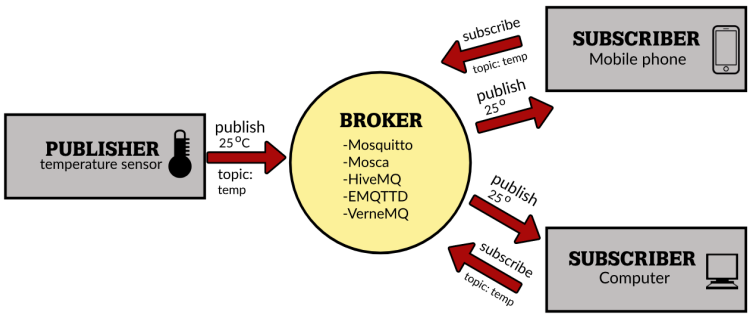

FIGURE 8.19: MQTT publish/subscribe model.

The topic of a given message can be considered as message subject. Basically, the topic name is an arbitrary UTF-8 encoded string. It can be also hierarchically structured using a forward slash (/) as a topic level separator.

[17]https://engineering.fb.com/android/building-facebook-messenger/
[18]https://docs.aws.amazon.com/iot/latest/developerguide/mqtt.html
[19]https://docs.openstack.org/infra/system-config/firehose.html
[20]https://www.home-assistant.io/components/mqtt/
[21]https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support
[22]http://envisense.org/floodnet/overview.htm

According to this rule, the topics have one or more topic levels. Exemplary three level topic can take form as: `myFirm/office1/temperature`. No prior initialization of the topic is required. It is not necessary to create a topic before publishing or subscribing to it. At least one character is needed to create a topic. The topic names are case-sensitive and can contain empty spaces.

Construction of a covert channel: We propose several covert channels for MQTT in (Velino et al., 2019), but the only channel for which a countermeasure variation was tested by the author of this thesis so far is described in the following. The robustness and bandwidth characteristics of the proposed channel can be found in the mentioned paper and are omitted here as they of limited relevance for the countermeasure variation.

The covert channel works as follows. All of the participants subscribe to previously agreed topics (Figure 8.20, step 1), which are enumerated and linked to secret bits. Then, if the covert sender wants to transmit a secret message, he publishes updates only to the topics that are mapped to bits which must be set to 1 in a certain message. For example, to transfer the 4-bit message 1011, the covert sender will publish status updates to the topics T_1, T_3 and T_4 (Figure 8.20, step 2), and all subscribers, together with the appropriate covert receivers, will receive these updates (Figure 8.20, step 3). From the presence/absence of the particular update on the agreed topic and previously agreed topic ordering scheme, covert receivers are then able to extract the hidden message. Our channel operates in 1-second intervals. This means that within one second, the appearance of the particular topics in publishing messages encodes that the related bits are set to "1", otherwise they are considered as "0". For example, if during the first interval, the topics T3, T4 but not T1 and T2, would appear, then the first four hidden bits would be "0011". If, during the next interval, the topics T2 and T3 would appear, the next four hidden bits would be "0110". Thus, two seconds form one ASCII byte (in this example case, it would be the byte "00110110", i.e., "6" in ASCII).

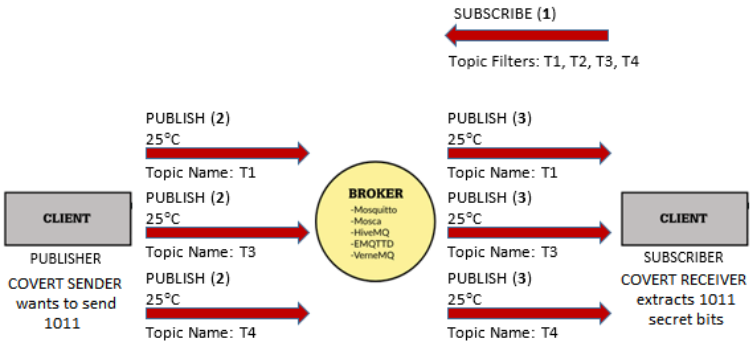

FIGURE 8.20: Indirect Covert Channel using Topic Ordering and Updates Presence/Absence.

Encoding of secret data	Number of legitimate and CC flows	Avg. number of CC's publish messages/flow	Avg. number of legitimate publish messages/flow	Sum MQTT publish messages
ASCII, 4 topics	100 each	2,180	2,377	455,700
AES, 4 topics	100 each	2,452	2,385	483,700
AES, 3 topics	100 each	1,940	1,880	382,000
AES, 2 topics	100 each	1,291	1,197	248,800

TABLE 8.12: MQTT traffic used to evaluate our approach

8.6.2 Countermeasure Variation

For the previously introduced covert channel, the hidden information is encoded in MQTT topics. For this reason, the appearance of topics in legitimate traffic must be distinguished from the appearance of topics in covert channel traffic. Again, we will try to apply the *compressibility* approach as described in the previous sections. Therefore, we generated 100 covert channel flows and compared its appearance of topics with 100 legitimate flows (Table 8.12).

First, we recorded all topics in the order of their appearance in a string S. For instance, if a flow would contain "Publish Message [T1]", followed by "Publish Message [T3]", "Publish Message [T4]" and "Publish Message [T1]", then our resulting string S would be "1341". Next, we compressed the string using a compressor $\mathfrak{I}$, i.e., $C = \mathfrak{I}(S)$ and compared its original length to the compressed length, i.e., S/C, resulting in a compressibility score κ.

Like for the previous covert channels, the compressibility score could not be applied directly to this new covert channel as we took topic order (instead of inter-arrival times) as an input, generated a different string and needed to find other window sizes and thresholds for κ. We determined the optimal string length for S by testing a set of parameters between 500 and 2,000 topics. In result, over all the scenarios that we test in the remainder of this section, the string length of 1,100 was a good choice. Finally, we tried to define an optimal combination of string size and κ-threshold to classify between legitimate and covert channel traffic.

8.6.3 Evaluation

When our channel encodes secret data in trivial ASCII traffic format using 4 topics (i.e., the presence of 4 topics represents 4 bits of the ASCII characters), the detection was providing excellent results (100% F-score). While such a channel might be the most suitable and most easy to implement by an attacker, one could also imagine that an attacker could encrypt the traffic in advance using AES. In the case of AES-encrypted traffic that is also encoded in 4 topics, the detectability decreases clearly (66.9% F-score). For comparison, the ROC curves of both variants of the covert channel are shown in Figure 8.21. The ASCII-encoded channel's AUC was 0.995 while the AES-encrypted channel's was 0.5996. Note, that we generated 100 legitimate and 100 covert flows for every scenario, each flow containing at least 1,100 topics.

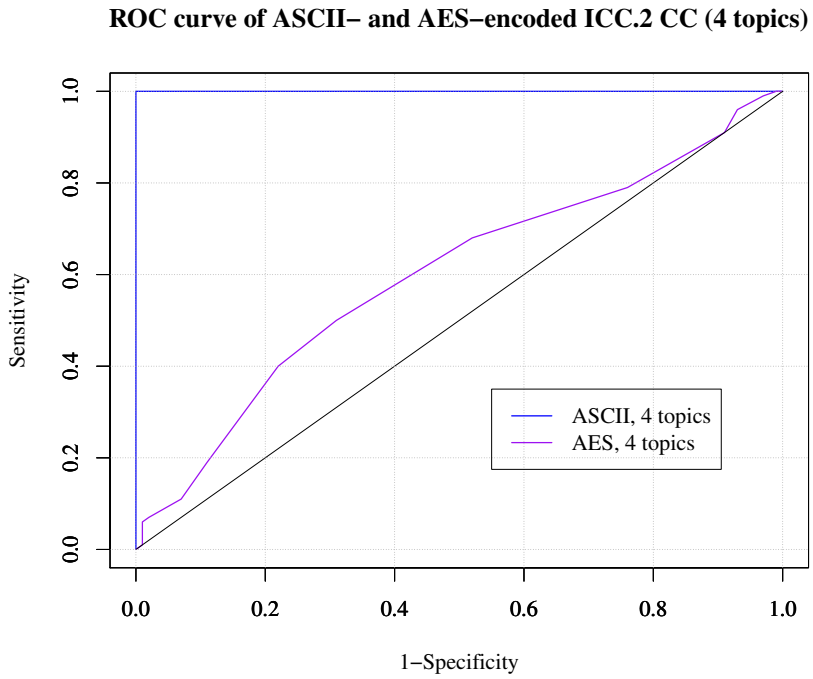

FIGURE 8.21: Detectability of MQTT-based covert channel (using 4 topics) with ASCII vs. AES-encrpted content.

Next, we wanted to examine whether an attacker could render the covert channel harder to detect if a different number of topics is used together with the AES-encrypted content. For this reason, we generated 100 legitimate and 100 covert channel-based flows for scenarios where 2, 3 and 4 topics are utilized. It turned out that in this case the detectability *slightly* decreases when fewer topics are used. The maximum F-score for 2 topics was 66.666% while it was 66.889% for 3 topics, and 66.89% for 4 topics. The differences are also visible in the ROC curves (Figure 8.22). Overall, the channel using only 2 topics could not be detected better than guessing; the accuracy was 50.5%, the precision 0.513% and the AUC was 0.4933. For the channel with 3 topics, accuracy increased to 54.5% and precision to 53.78%, which is still hardly detectable, while resulting in an AUC of 0.5472. Depending on the used κ threshold, the channel with 4 topics could be detected either with 52.5% accuracy (but 85.71% precision) or with 59.5% accuracy (61.73% precision).

Afterwards we analyzed the optimal detection thresholds for κ (Figure 8.23) under consideration of maximizing the F-score. The optimal thresholds were similar in case of the AES-encoded channels. For 2 topics, the best performing values were between $\kappa = 3.0$ and $\kappa = 6.5$ (all providing the same F-score of 66.66%). For 3 topics, the best performing values were between $\kappa = 3.0$ and $\kappa = 5.15$ (optimal: $\kappa = 4.95$ and $\kappa = 5.0$) and for 4 topics, best values were between $\kappa = 3.0$ and $\kappa = 4.4$ (optimal: 4.3). Overall, the threshold $\kappa = 4.3$ provided the best *average* F-scores to detect all three AES channels (all F-scores between 66.666% and 66.89%). However, the ASCII channels' best detectability was given for thresholds between 4.8 and 5.25 (all close to or exactly 100% F-score), while $\kappa = 4.3$ would only provide an

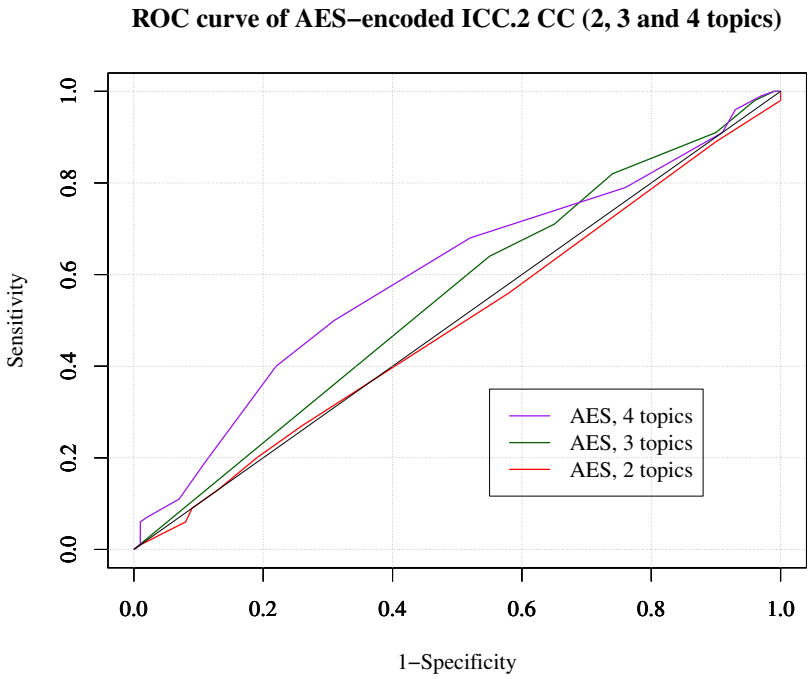

FIGURE 8.22: Detectability of the MQTT-based covert channel that uses AES-encryption in combination with 2, 3 and 4 topics.

F-score of 66.89%.[23]

Finally, when realistic conditions are expected, then only a fraction of the flows might contain a covert channel. For this reason, thresholds should be selected in a way that would keep the false-positive-rate (FPR) at a minimum. As shown in Figure 8.24, the FPR for the ASCII and the AES channel with 4 topics drops close to zero at $\kappa = 4.75$, rendering this threshold suitable for realistic conditions where the number of false-alarms must be minimized. The F-score for the ASCII channel with 4 topics and $\kappa = 4.75$ is 99.5% while the FPR is 1%, the F-score even increases to 100% with an FPR of 0% for $\kappa = 4.8$.

However, for the AES channel with 4 topics, the F-score for the same threshold of 4.75 is only 12.84%. There is no threshold available to detect the AES 4-topic channel with an F-score of at least 50% while maintaining FPR below 30%, rendering this channel not detectable under conditions where the fraction of covert channel traffic is low. The same applies to the AES channel with 3 and 2 topics (but with worse results). We can thus conclude that only the ASCII channel can be detected under condition where the fraction of covert channel traffic is either high or low while the AES-based channels are not detectable under realistic conditions.

We expect malware authors to implement simple versions of a covert channel first (as can be seen in the cases of currently known stegomalware that

[23]The κ values of legitimate and AES-encoded covert channel traffic overlap clearly. Thus, we decided to apply an interval (similarly like done for scenario 1 of this chapter) instead of only one threshold for κ. However, these attempts did not improve detection results.

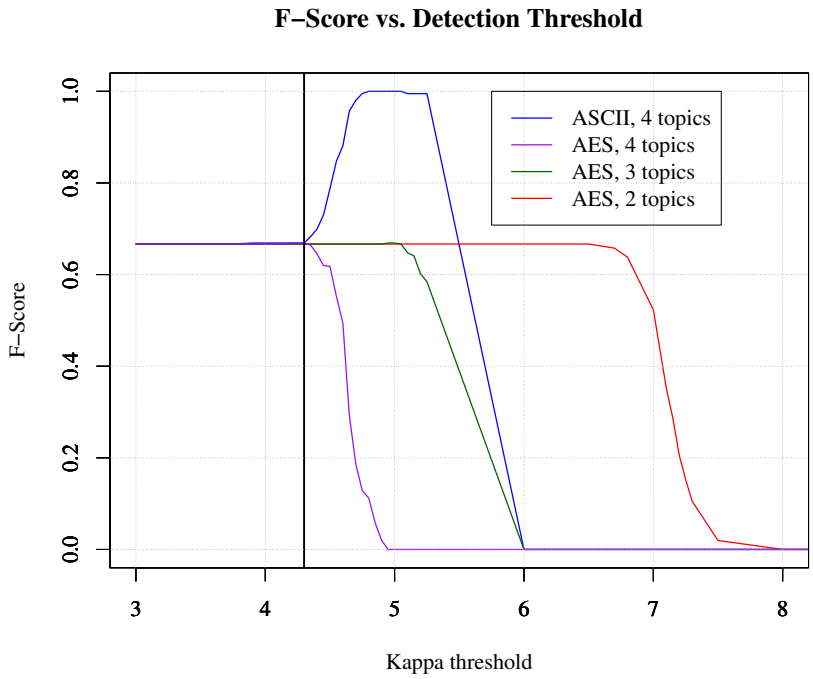

FIGURE 8.23: Optimization problem for determination of the optimal threshold – 4.3 performs best, overall, but higher thresholds benefit specific other channels.

apply usually trivial methods) but it would be enough to encrypt the hidden content to render this covert channel undetectable when solely the compressibility of topic appearances is analyzed. Thus, we conclude that this countermeasure variation solely works for the ASCII version of the channel.

The value modulation pattern is a rather heterogeneous pattern, rendering the options to realize a covert channel manifold. Thus, we cannot conclude that the compressibility score performs better or worse for other covert channel techniques that represent a value modulation. This is especially the case since the MQTT protocol's behavior is rather specific to publish-subscriber protocols.

8.7 Additional Countermeasure Variations

In addition to the previously shown scenarios, we also performed a countermeasure variation for the *Artificial Re-transmission* pattern using the ϵ-similarity and the compressibility score (Zillien and Wendzel, 2018). In particular, the ϵ-similarity was leading to an accuracy of 97.59%, a precision of 99.35%, and a recall of 96.25% (also sophisticated channels were detectable). The compressibility score resulted in a less promising accuracy of 78.9%, a precision of 83.95%, and a recall of 85%.

Additionally, we used the so-called *regularity* metric (a simple metric based on standard deviation) to detect the first scenario's *Size Modulation* pattern (Wendzel et al., 2019). High-quality results were especially achieved for

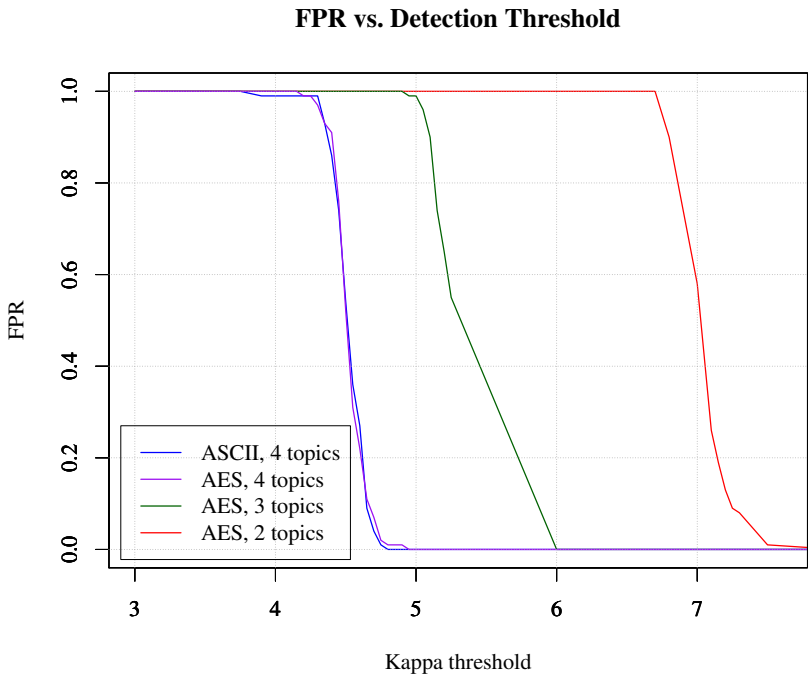

FIGURE 8.24: False-positive-rate of the MQTT-based covert channel, depending on encoding and number of utilized topics.

two-symbol channels with 50 and 60 bytes (99.2% accuracy and 93.4% precision) as well as for channels with a difference of only 1 byte in their size, even if they contained more than two symbols. For other packet size differences, accuracy and precision were both around 80%.

8.8 Discussion and Conclusion

We have shown that countermeasure variation for network covert channels is feasible. Therefore, it is necessary to *transform* a detection method, i.e., to adapt it so that it works with another hiding pattern. We exemplified the feasibility of countermeasure variation by transforming the compressibility score and the ϵ-similarity that were originally introduced to detect covert timing channels so that they can be applied to covert channels of the Size Modulation, PDU Order and Value Modulation patterns.

Size Modulation: After applying countermeasure variation to the *compressibility* metric, we were able to detect two-symbol covert channels utilizing a 1-byte packet size difference with 97.17% accuracy and 95.82% F-score, given that suitable detection intervals were selected. The detectability slightly decreased down to 93.59% accuracy and 91.21% F-score for larger packet size differences. For a mixture of different two-symbol covert channels, we could achieve an accuracy of 91.40% and an F-score of 92.07%. Covert channels with either 3, 4, *or* 8 different symbols were well-detectable with different (non-overlapping) thresholds. These thresholds were not able to effectively detect a *mixture* of these channels. However, when a

wider threshold was applied, a balanced mixture of covert channels with 3, 4, and 8 symbols became detectable with an accuracy of 92.83% and an F-score of 93.28%.
The countermeasure variation of the *ϵ-similarity* provided high-quality results for two-symbol covert channels under narrow limits (0.34%/0.35% of all values below $\epsilon = 0.1$ with an accuracy of 97.8%, a precision of 95.8%, and a recall of 100%). Out of these limits, no detection was possible. The approach did not provide useful results when the channels had more than two symbols.

PDU Order: Our results have shown that the detectability of PDU order channels depends on the number of utilized PDUs and different thresholds should be applied for these channels. Covert channels utilizing 3 or 4 PDUs were detectable with more than 99.5% accuracy and F-score. The low false-positive rates of <1% and <0.1%, respectively, for these channels renders our approach even useful for scenarios with high volumes of traffic. This is important for a real-world application since these two channels can transfer the more information per packet than a channel using only 2 PDUs. Such channels with 2 PDUs were detectable with an accuracy and F-score of 94.51% while their optimal threshold was linked to a false-positive rate of approx. 5.1%, rendering our approach not suitable for big-data scenarios (except for targeted scenarios with a low traffic volume or when a lower true-positive rate would be acceptable).

Value Modulation: We were able to detect a simple ASCII-encoded version of an MQTT-based covert channel that performs a value modulation when we applied the compressibility score. However, the results were unacceptable (less than 67% F-score) in case of AES-encoded values, which basically represent the scenario used for the two previous patterns. The value modulation pattern is a rather heterogeneous pattern, rendering the options to realize a covert channel manifold. Thus, we cannot conclude that the compressibility score performs better or worse for other covert channel techniques that represent a value modulation.

In future work, we plan to conduct additional tests under different network conditions and with new metrics, such as the regularity metric that we already applied in follow-up work (Wendzel et al., 2019). As shown in (Zillien and Wendzel, 2018), countermeasure variation also works for additional patterns. This motivates the analysis of further patterns (e.g., Random Value pattern or Add Redundancy pattern) in follow-up publications.

Chapter 9

Extensions of the Original Pattern Taxonomy

Abstract In this chapter, we analyze the current state of hiding patterns and we further improve their taxonomy. In order to more thoroughly characterize and understand data hiding methods applied to communication networks we propose to distinguish between sender-side and receiver-side patterns. Additionally, we show how hiding patterns can be utilized to conveniently describe the realization of the distributed network covert channels.

Originally published: Mazurczyk, Wendzel, Zander, Houmansadr, and Szczypiorski (2016a, Chapter 3); Mazurczyk, Wendzel, and Cabaj (2018)

9.1 Introduction

In this chapter, we analyze the key aspects of hiding patterns and the current state of the taxonomy in the domain. However, the main contributions of this chapter are that we show how this concept can be further extended by modifying the pattern-analysis process and extending the current taxonomy with new patterns. By taking into account more details on the hiding method's inner workings we hope that the resulting pattern categorization will contribute to a better understanding of the nature of network covert channels. Moreover, we also introduce and describe a pattern-based classification of *distributed* network covert channels.

The rest of this chapter is structured as follows. Section 9.2 introduces fundamentals and related work on hiding patterns. We discuss limitations of the current patterns approach in Section 9.3. Section 9.4 introduces our improved taxonomy, a process for pattern-analysis as well as new patterns dedicated to the payload field and our pattern-based categorization of distributed network covert channels. Finally, Section 9.5 concludes our work and provides an outlook on future research directions.

9.2 Fundamentals

To aid the understanding of information hiding methods, an analysis of the existing network covert channels and corresponding protocols should be performed. Hiding patterns provide an abstract and hierarchical view on these methods and their utilization in combination with network protocols.

As a starting point, we utilize the patterns presented in Chapter 4. In (Wendzel et al., 2015) and (Wendzel et al., 2016), we evaluated more than 130 existing network covert channel techniques from past decades and extracted abstract patterns from these techniques. We were able to represent all techniques by (only) 11 patterns, which were arranged in a hierarchical catalog described using *Pattern Language Markup Language* (PLML). While later work in (Mazurczyk et al., 2016a) modified and extended our patterns, the core part of the hierarchy and several patterns remained (colored in white and light-gray in Figure 9.1). Later modifications and extensions by (Mazurczyk et al., 2016a) are colored in darker gray in Figure 9.1. The latest description of all patterns shown in this figure is presented briefly in Table 9.1.

As it can be seen in Table 9.1, a hiding pattern's description is written in an abstract manner so that one pattern can be used to describe multiple hiding techniques at the same time. For instance, "modulate the least significant bits of a protocol field" is a very brief description of many published hiding methods which utilize the least significant bits of fields in arbitrary network protocols.

The above-mentioned classification is carrier-oriented and a "carrier" is defined as one or more overt traffic flows that pass between the covert sender and the covert receiver, consisting of protocol data units (PDUs, e.g. frames or packets). Typically, the carrier can be multi-dimensional, i.e. it

Pattern Name	Pattern Description
Rate/ Throughput	The covert channel sender alters the data rate of traffic from itself or a third party to the covert channel receiver.
Inter-packet Times	The covert channel alters timing intervals between network PDUs (interarrival times) to encode hidden data.
Message Timing	Hidden data is encoded in the timing of message sequences, e.g. acknowledging every n'th received packet or sending commands m times.
Artificial Loss	The covert channel signals hidden information via artificial loss of transmitted messages (PDUs).
Frame Collisions	The sender causes artificial frame collisions to signal hidden information.
Temperature	The sender influences a third-party node's CPU temperature, e.g. using burst traffic. This influences the node's clock skew. The clock skew can then be interpreted by the covert receiver by interacting with the node.
Retransmission	A covert channel retransmits previously sent or received PDUs.
Message Ordering	The covert channel encodes data using a synthetic PDU order for a given number of PDUs flowing between covert sender and receiver.
Size Modulation	The covert channel uses the size of a header element or a PDU to encode a hidden message.
Sequence Modulation	The covert channel alters the sequence of header/PDU elements to encode hidden information. This pattern divides further into: P2.a. Position and P2.b. Number of Elements patterns.
Add Redundancy	The covert channel creates new space within a given header element or within a PDU in which to hide data.
Random Value	The covert channel embeds hidden data in a header element containing a (pseudo-)random value.
Value Modulation	The covert channel selects one of the n values that a header element can contain to encode a hidden message. This pattern divides further into: P6.a. Case Pattern and P6.b. Least Significant Bit (LSB) patterns.
Reserved/ Unused	The covert channel encodes hidden data into a reserved or unused header/PDU element.

TABLE 9.1: Information hiding patterns as introduced in (Wendzel et al., 2015) and updated in (Mazurczyk et al., 2016a).

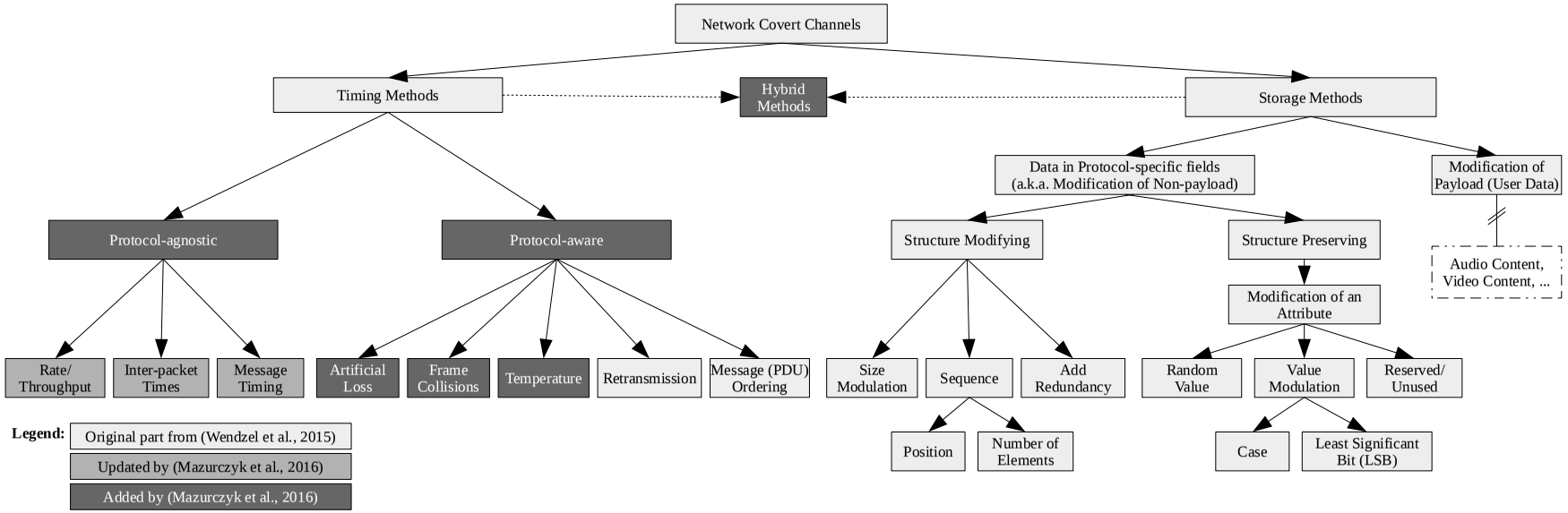

FIGURE 9.1: Hiding patterns as introduced in (Wendzel et al., 2015) and updated in (Mazurczyk et al., 2016a).

offers many opportunities "places" or "events" for hiding data (called sub-carriers). As in other network covert channel categorizations the two main groups of methods are (Figure 9.1):

- *storage methods*: a class of network steganography methods that modify the "places" (sub-carriers) in a carrier to create a storage covert channel. These techniques hide information by modifying e.g. protocol fields, such as unused bits of a header.
- *timing methods*: a class of network steganography methods that modify the timing of "events" of a carrier to create a covert channel. These techniques hide information, e.g. in the timing of protocol messages or packets.

Some important changes have been introduced in (Mazurczyk et al., 2016a) when compared with original categorization from Chapter 4. These include:

- defining 14 patterns (8 timing patterns and 6 storage patterns), compared to 11 patterns (4 timing and 7 storage) proposed originally. Note that the increased number of hiding patterns is mainly caused due to adding new layer of classification in (Mazurczyk et al., 2016a) for timing patterns which have been divided into "protocol agnostic" or "protocol aware" groups.
- the pattern 'PDU Corruption/Loss Pattern' has been removed from the storage patterns and instead the 'Artificial Loss' pattern which full name is 'Artificial Message/Packet Loss' and the 'Frame Collision' pattern have been added to the list of timing patterns.
- A few patterns have been slightly modified/renamed.

Chapter 4 introduced also several other concepts which explain suitably some network covert channels' phenomena, i.e. pattern variation, pattern combination, and pattern hopping. These concepts will be extended in this chapter and are briefly explained below:

First, *pattern variation* is a transformation-like approach for covert channels. The utilized network protocol is defined as the pattern's context. Therefore, a pattern's application can change from one network protocol to another – without redesigning the most important aspects and inner workings of the hiding technique itself. Next, *pattern combination* allows the use of multiple patterns at the same time (within the same carrier, e.g. by modifying many sub-carriers at once). This is typically performed to increase available steganographic bandwidth – thus in short it is a parallel utilization of multiple network covert channels simultaneously. Finally, *pattern hopping* varies the use of patterns over time – usually it is applied in order to increase stealthiness. This can be briefly summarized as a sequential utilization of various network covert channels in time using different (sub-)carriers.

It must be also noted that in the reminder of this chapter we will rely on the unified description for network information hiding methods introduced in Chapter 6. This chapter has been the first attempt to standardize the description of network covert channels which is suitable, e.g. to assess their novelty and impact of the method on the state-of-the-art. In Chapter 6,

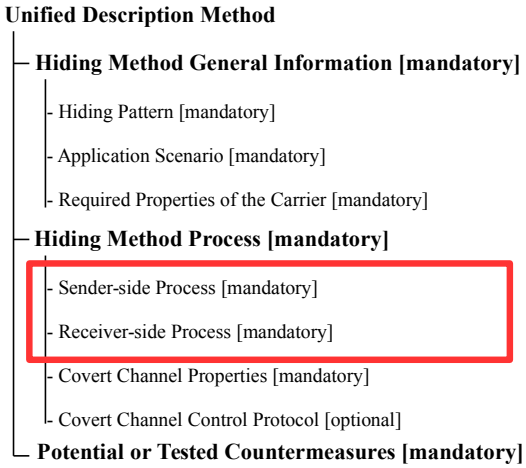

FIGURE 9.2: The unified description structure for data hiding methods as introduced in (Wendzel et al., 2016).

the proposed description of data hiding methods was split into three categories: *(i)* general information about the hiding method; *(ii)* description of the hiding process, and *(iii)* potential or tested countermeasures. The first two categories comprise sub-categories and each (sub-)category can be mandatory or optional (Figure 9.2).

The category "hiding method general information" consists of a link to existing network hiding patterns. It also includes a discussion of the application scenario and requirements of the carrier. From the perspective of this chapter the most important category, i.e. "hiding method process", is split into four parts: the sender-side and the receiver-side description of the hiding method, the details of the covert communication channel, and the description of an associated covert channel control protocol (if applicable). The third category discusses both, potential and evaluated countermeasures, including those that detect, limit or prevent the particular hiding method's use. In the following we will reference to the fragments of this unified description when it comes to the pattern categorization.

9.3 Analysis of the Existing Taxonomy

Our analysis has shown that the current information hiding patterns approach can be further extended to include the following aspects:

- *Incorporate More Details on Data Hiding Methods:* The key criterion of the current pattern taxonomy for deciding which pattern an analyzed method represents is to determine how the secret data is encoded. Thus, due to this it is omitting some details on how the data hiding method works (from the sender-side and receiver-side process – this will be shown further in the next sections). This "flattens" the description of the inner workings of the data hiding methods and thus may prevent that *all* details of a hiding method are considered. A more thorough patterns grouping is desired to more accurately categorize existing network steganography methods.

- *Support Hybrid Patterns:* For some cases it is difficult to assess whether the analyzed method is storage, timing or hybrid – a clearer distinction and unambiguous formula to deduce this is desirable.
- *Multi-Packet and -Flow Characteristics Support:* The current categorization makes no clear distinction between hiding methods that are focusing on a single packet or multiple packets. Also, there is no clear distinction between single- and multi-flow methods. For example, consider a covert channel that modulates IPv4 ToS values in such a way that the sequence of ToS values from the consecutive packets is interpreted as a single secret data bit – currently such a method does not match any hidden data pattern. Moreover, some hiding methods such as (Luo et al., 2007) utilize multiple flows. It is thus beneficial to make the original pattern descriptions more generic, i.e. less dependent on specific units such as PDUs or packets.
- *Coverage of Sophisticated Hiding Methods:* It is not exactly clear whether more advanced network steganography concepts like inter-protocol steganography (Jankowski et al., 2013), protocol switching covert channels (Wendzel and Zander, 2012), multi-level steganography (Fraczek et al., 2012a), adaptive covert communication (Yarochkin et al., 2008), etc. can be accurately expressed using current pattern categorization. Pattern combination, pattern hopping and pattern variation are means to represent them, but not to the full extent.
- *Influence on Payload:* In the original design decision of the pattern-based approach, arbitrary content, e.g. digital files, were considered as part of digital media steganography instead of network information hiding. However, in some cases, such as in VoIP steganography, where there are data hiding methods that affect the payload field, it can be helpful to have a taxonomy that covers also the transmitted payload. In principle the patterns should be analogous as they too adhere to the storage group.
- *Distinction Between Secret Data Embedding and Transfer:* It is also worth to emphasize that from the pattern-based countermeasures perspective it is more important to know which pattern represents the covert technique *within* the communication channel. It must be noted that in particular the information hiding patterns used at the sender-side process to embed secret data may not exactly represent themselves in the same while traversing within the hidden data carrier through the communication network.
- *Embrace PDU Corruption Pattern:* As mentioned, in (Wendzel et al., 2015) 11 (4 timing and 7 storage) patterns have been defined while in (Mazurczyk et al., 2016a) there are 14 (8 timing and 6 storage) patterns. However, the pattern 'PDU Corruption/Loss' has been removed from the storage patterns group by (Mazurczyk et al., 2016a). In fact, it is our belief that it is beneficial that the 'Artificial Message/-Packet Loss' pattern has been added into timing patterns but still the 'PDU Corruption' pattern should be considered in storage scenarios.

Based on the above-mentioned points, we describe how we envision enhancements to the current information hiding patterns concept in the next section.

9.4 Extension & Modification of the Patterns Approach

In this section, we present the proposed modification for the original information hiding patterns concept which can help in deriving further insights into understanding the nature of various types of network covert channel techniques. More specifically, in Section 9.4.1 we propose how the original pattern approach can be extended in order to include the sender-side and receiver-side processes which influences both pattern creation process and covert techniques categorization. Next, in Section 9.4.2 we propose new patterns applicable to the payload field. Finally, in Section 9.4.3 we discuss the *distributed* network covert channels and how the information hiding patterns concept can be used to conveniently describe them.

9.4.1 Process for Pattern-Application Analysis

Considering the arguments from Section 9.3, we propose an approach based on (Wendzel and Palmer, 2015), which describes how to determine the novelty of a new hiding technique and whether a hiding technique actually represents a *new* pattern, or not. Instead, our goal is to gain additional insights into the inner-workings of the data hiding method, i.e. we do not *replace* the original approach.

In the current categorization, authors of a new data hiding technique first describe their technique, e.g. informally or using (Wendzel et al., 2016). Then, based on how the secret data is *embedded* one pattern is selected that represents the hiding method. Therefore, authors first decide whether the hiding method is storage or timing, then, whether it is protocol-aware or agnostic (timing channel) or structure-preserving or structure-modifying (storage channel). If a hiding method does *not* fit into the current pattern representation, it is considered a *new* pattern which can be added to the taxonomy. The related decision-making process can be found in (Wendzel and Palmer, 2015).

We propose a similar but modified version of this approach. However, as mentioned, our approach targets a different goal, namely to derive *more insights* related to the information hiding method itself. It must be noted that we do not focus only on how the secret data for a certain data hiding method is embedded (which is only a part of the sender-side process) but instead we want to detail both the complete sender- and receiver-side processes and represent them with patterns (and for this purpose, we "borrow" the already existing patterns.

In our proposed approach, the known hiding patterns of existing publications and websites, e.g. (Wendzel et al., 2015; Mazurczyk et al., 2016a) or *https://ih-patterns.blogspot.com*, which are tagged as storage or timing patterns, are taken into account. Then for the hiding method that needs to

be described using the network covert channel patterns approach, the corresponding patterns for both, sender- and receiver-side processes are selected. Finally, based on the result and depending on what types of patterns have been assigned to the method, the *method itself* is concluded as a storage, a timing or a hybrid method – this selection process is explained in the details below.

The described improved approach which aims to derive more insights from the data hiding methods using pattern approach allows to repaint the categorization from Figure 9.1. However it must be noted that in the modified approach we categorize *network covert channel patterns* and not data hiding *methods*. Thus, we start the derived classification from the network covert channel patterns which are then divided into timing and storage ones (Figure 9.3). Afterwards, each of the methods that needs to be evaluated is assigned with at least one or more patterns to its sender- and receiver-side processes separately (for each side at least one or alternatively more patterns must be selected).

It must be also noted that using this approach it may be possible to evaluate in greater detail which patterns are most often used jointly only at the sender-side process (as more than one pattern can be assigned) or only at the receiver-side process, or finally which patterns typically coexist at the sender-side and the receiver-side processes. This can be achieved by performing a thorough analysis of network covert channels defined in the literature (however, due to space limitation it will be not part of this chapter). In result of such an analysis this can lead to the identification of potential relationships between defined patterns, i.e. whether for some of them it is "easier" to coexist with other patterns within the data hiding method (as in the case of the extended approach the sender and the receiver processes can be investigated separately or jointly).

But more importantly, it is also possible to investigate whether besides of joint patterns utilization (at the sender-side, receiver-side or both sides), other pattern mixes are also possible. For example, consider Method 4 in the Figure 9.3. It is characterized by the patterns *Retransmission* and *Size Modulation*, which makes it a hybrid method. However, the question arises whether is would be possible to construct a data hiding method that apart from these two patterns utilizes e.g. *Message (PDU) Ordering* pattern and how this will impact its properties.

In result, new, previously unknown network information hiding methods or improved versions of existing ones can be designed and developed and relationships between the existing patterns can be investigated and determined. It must be noted that using the existing pattern classification it was possible to assign only a single pattern for a certain hiding method which corresponds best with the secret data embedding process. However, in the extended approach (which is different when compared to the original concept) it is possible to:

- assign more patterns to the sender-side process if it is required in order to express to a full extent how the sender-side of the hiding method operates,
- include also the receiver-side process and its corresponding patterns.

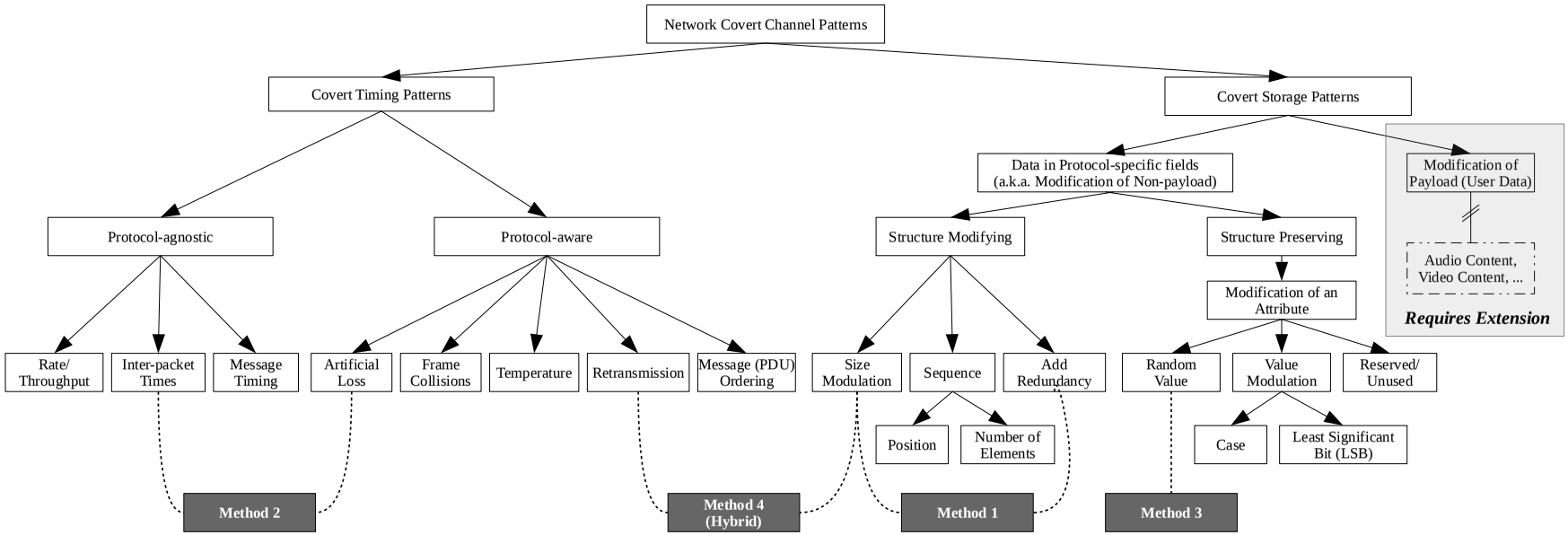

FIGURE 9.3: Improved aspects of the existing pattern-based taxonomy.

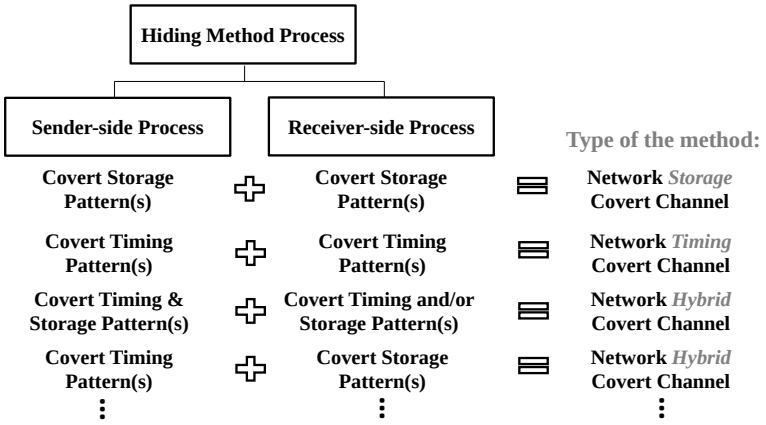

FIGURE 9.4: Improved process to decide on the network covert channel type based on the assigned patterns.

Such an approach may not only help to better understand the nature of the network covert channels and their creation process, but it can also provide new insights into how to construct more efficient and effective detection solutions. This can be achieved by designing and developing detection methods, so they precisely will be looking for the specific artifacts related to the representation of the certain patterns in the communication channel (and/or e.g. the presence of their coexistence).

Finally, each method based on the selected patterns for the sender- and for the receiver-side processes is assigned to one element of the group {*storage*, *timing*, *hybrid*}. This is done as illustrated in Figure 9.4. In principle, if both the sender- and the receiver-side processes are characterized with homogenic (only storage or only timing) patterns then the method is concluded as storage or timing. If there is heterogeneity across patterns that the method uses, i.e. storage and timing methods are mixed within the sender- and/or the receiver-side processes then it is concluded as a hybrid technique.

To present how the proposed extended patterns' classification approach is functioning for some of the existing network steganography techniques, we have chosen seven different state-of-the-art network covert channels to demonstrate how they fit into our categorization (Figure 9.5). For example, for a simple network covert channel which in order to conceal data utilizes Type of Service field from the IPv4 header (Handel and Sandford, 1996), the sender- as well as receiver-side processes use the same pattern, i.e. *Reserved/Unused*, thus as both processes are assigned with the storage pattern then the method is concluded as storage. For the work related to modifying delays between the consecutive packets within the data stream (Berk et al., 2005) for both sender- and receiver-side processes the pattern *Inter-arrival time* is an obvious choice thus this technique is deemed as timing method. However, when we consider a more complex method like LACK (*Lost Audio Packets Steganography*) (Mazurczyk and Lubacz, 2010) then the situation is a bit different. As LACK operates by using intentionally delayed voice packets and replacing the original payload of these packets with secret data thus at the sender-side process two patterns must be selected –

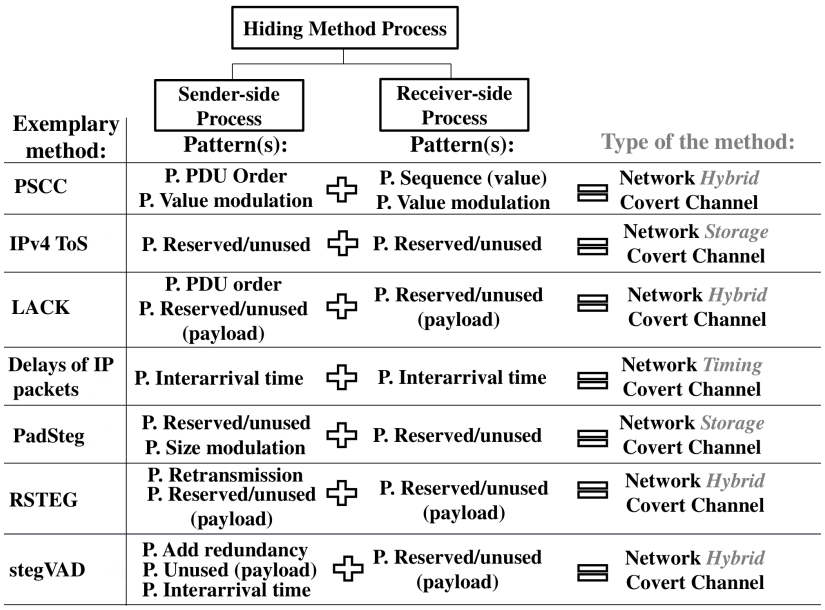

FIGURE 9.5: Classification of the exemplary network covert channels based on the assigned patterns.

one storage (*Reserved/Unused*) and one timing (*PDU Order*), whereas when considering the receiver-side process the chosen pattern is only storage one (*Reserved/Unused*) – as at the covert receiver every incoming packet's payload, regardless of its order, is probed for the existence of the hash which will indicate presence of secret data. Therefore, the method is concluded to be hybrid. It is worth emphasizing that if we consider the original pattern approach (which as mentioned relied only on assigning pattern(s) based on how/where secret data is embedded) then LACK method would be only characterized by the storage *Reserved/Unused* pattern. This proves that the extended pattern approach proposed in this chapter allows to characterize the data hiding methods in greater detail by including more information on inner workings of the information hiding technique.

9.4.2 Introduction of Additional Patterns

As already mentioned, the current pattern-based categorization of (Wendzel et al., 2015; Mazurczyk et al., 2016a) makes a distinction between patterns applied to user-data (within the payload field) and protocol specific data (control information: headers, padding, etc.). In principle, all these patterns adhere to the storage group, i.e. modification of the certain "locations" of the carrier. However, in the original publications on hiding patterns, this distinction was made based on the idea of Fisk et al. (Fisk et al., 2003) to separate structured (machine-readable) content from non-structured (human-readable) content, such as images. This means that in several cases similar rules apply to modify these fields (because structured data follows rules, e.g. protocol headers are built similarly to formal grammar) and to the data that they store. Obviously the most significant difference lays in the dissimilarities between the control information carried within protocol headers/padding and user-data transferred within the payload field. Thus, to

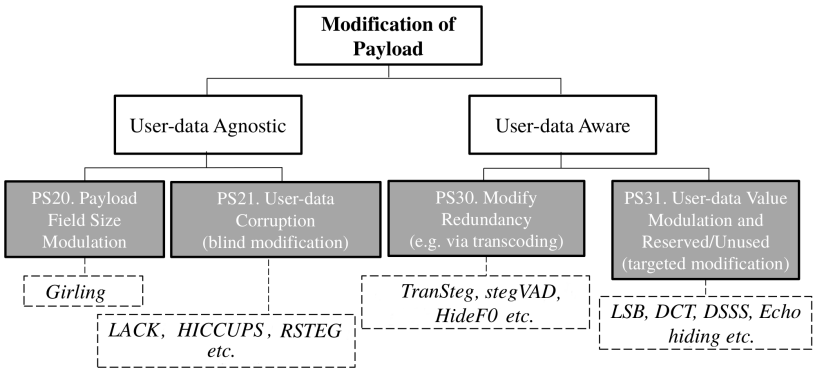

FIGURE 9.6: Classification of the network covert storage channels for the payload field and the corresponding patterns.

fill this gap and by considering current research efforts in this area, we propose to extend the current taxonomy as shown in Figure 9.6.

Network covert channels that modify the payload field and its content have been divided based on whether the characteristic of user-data is taken into account into: *(i)* user-data agnostic and *(ii)* user-data aware. In each of the two groups two patterns have been identified, which we describe in the same way as the patterns were originally described in (Wendzel et al., 2015) using a subset of the *Pattern Language Markup Language's* (PLML) attributes:

PS20. Payload Field Size Modulation
Illustration: This pattern uses a size of the payload field of a flow's PDUs/ messages to encode the hidden message. This pattern is a variant (child) of the pattern *P1. Size Modulation* of (Wendzel et al., 2015) which has been already defined for the modification of the non-payload branch of storage methods (confirm Figure 9.1).
References: PS1. Size Modulation
Context: Network Covert Channel Patterns → Covert Storage Channel Patterns → Modification of Payload → User-data Agnostic
Evidence:
1. Modulate the size of the data block field in Ethernet frames (Girling, 1987).
2. Any other method that modulates the size of the payload field in any network protocol.

PS21. User-data Corruption
Illustration: This pattern is related to the cases when steganographic methods do not take into account what kind of user-data is carried within a payload field and/or what its characteristic is (blind modification). It can be applied to single PDUs or to multiple PDUs (a flow). This typically happens if parts of (or the whole) user-data is replaced with secret bits and thus the user-data is corrupted/lost. This pattern is similar to the pattern *PDU Corruption* defined in the original pattern categorization of (Wendzel et al., 2015).

Context: Network Covert Channel Patterns → Covert Storage Channel Patterns → Modification of Payload → User-data Agnostic
Evidence:
1. Replace the user-generated data in the payload field with secret data in intentionally lost voice packets of the IP telephony call (Mazurczyk and Lubacz, 2010).
2. Replace the user-generated data in the payload field with secret data in retransmitted TCP segments (Mazurczyk et al., 2011).
3. Replace the user-generated data in the payload field with secret data in intentionally corrupted IEEE 802.11 frames (Szczypiorski, 2012).

PS30. Modify Redundancy
Illustration: This pattern is used when it is possible to exploit the redundancy of the user-data by means of transforming them in such a way that a free space for secret data is obtained (e.g. by means of transcoding). This pattern is a bit similar to the pattern *Add Redundancy* defined in (Wendzel et al., 2015) but can also decrease redundancy and is applied to payload instead of meta-data.
Context: Network Covert Channel Patterns → Covert Storage Channel Patterns → Modification of Payload → User-data Aware
Evidence:
1. Compress existing user-data in order to make a space for secret data (Mazurczyk et al., 2014).
2. Transform the VAD-enabled IP telephony voice stream into non-VAD one and fill the gaps using artificially generated RTP packets containing secret data (Schmidt et al., 2017).
3. Approximate the F0 parameter of the Speex codec which carries information about the pitch of the speech signal and use the "saved" space for secret data (Janicki, 2016).

PS31. User-data Value Modulation and Reserved/Unused
Illustration: Characteristic features of user-data can be utilized to store secret information. This includes applying methods like LSB modification to speech samples or digital images carried within the payload field. Compared with previous patterns this is a targeted modification. This pattern is analogous to the combination of the patterns *Value Modulation* and *Reserved/Unused*, but applied to payload.
Context: Network Covert Channel Patterns → Covert Storage Channel Patterns → Modification of Payload → User-data Aware
Evidence:
1. Encode a stream of information by spreading the encoded data across as much of the frequency spectrum as feasible (e.g. DSSS) (Bender et al., 1996b).
2. Embeds secret data into a carrier audio signal by introducing an echo (a.k.a. echo hiding) (Bender et al., 1996b).
3. Replacing the least significant bit of e.g. each voice sample with secret data (LSB) (Bender et al., 1996b).

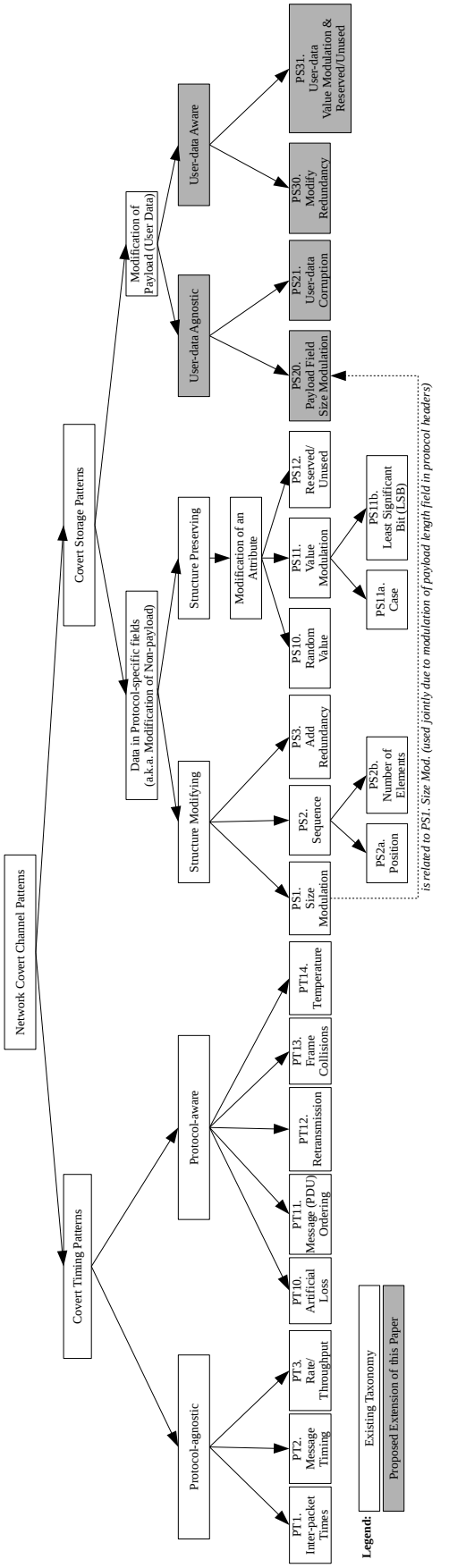

FIGURE 9.7: Classification of network covert channel patterns.

As it is visible above, the identified patterns have mostly a number of examples in the state-of-the-art publications (Figure 9.6). Every newly defined pattern corresponds to the patterns that have been already defined in the *non-payload* branch of the original classification.

Finally, the complete picture of the extended information hiding patterns classification is illustrated in Figure 9.7 and the corresponding descriptions of all defined patterns which include also potential multi-packet/multi-flow characteristics of some data hiding methods are enclosed in Table 9.2.

9.4.3 Distributed Covert Channel Realization

In Chapter 4, we defined three concepts which can be used to explain suitably some of the existing network covert channels' phenomena, i.e. pattern variation, pattern combination and pattern hopping.

The above-mentioned concepts are especially suitable and important when trying to depict, explain, and analyze the realization of *distributed* network covert channels. We define a *distributed covert channel* as a network covert channel that spreads secret data among multiple flows/protocols/hosts or uses multiple patterns within the same flow or PDU for the hidden data exchange. In contrast, the typical (undistributed) network covert channel is a storage or a timing channel that uses PDUs of a single flow/protocol with only one hiding pattern in order to embed secret data.

In Figure 9.8 we have illustrated that these three pattern concepts practically exhaust possibilities for distributed network covert channel realization. While explaining these concepts we apply the terms of *spatial*, *temporal*, and *transform domains* which are "borrowed" from the digital media steganography research area (Subhedar and Mankar, 2014) and which helps to described and define them better.

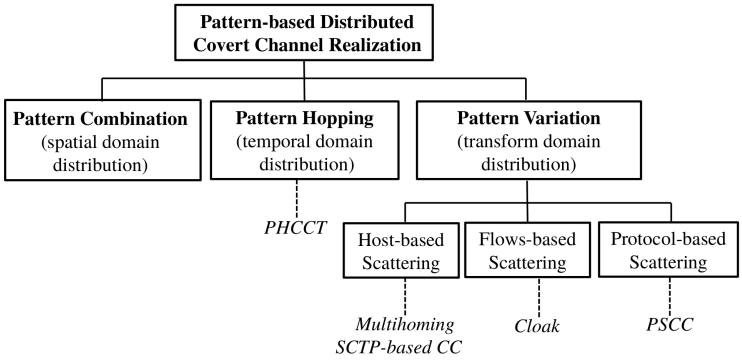

FIGURE 9.8: Classification of pattern-based distributed covert channels.

The first group i.e. *pattern combination* is related to the distribution of secret data in a *spatial domain*. This means that many patterns are utilized in parallel for the same hidden data carrier e.g. by modifying many of its subcarriers or using several carriers at once. This includes the case when the hybrid data hiding methods are used (cf. Figure 9.1) as well as the case of simultaneous utilization of multiple network covert channels at once. Consider an example of HTTP traffic (e.g. web browsing) where three separate

Pattern Name	Pattern Description
PT1. Inter-packet Times	The covert channel alters timing intervals between network messages of a flow (interarrival times) to encode hidden data.
PT2. Message Timing	Data is hidden in the timing of message sequences within a flow, e.g. acknowledging every n'th received message or sending commands m times.
PT3. Rate/ Throughput	The covert channel sender alters the data rate of a flow from itself or a third party to the covert receiver.
PT10. Artificial Loss	Hidden information is signaled via artificial loss of a flow's transmitted messages, e.g. by frame-corruption or message drop.
PT11. Message Ordering	The covert channel encodes data using a synthetic message order in a flow.
PT12. Retransmission	A covert channel retransmits previously sent or received messages of a flow.
PT13. Frame Collisions	The sender causes artificial frame collisions to signal hidden information.
PT14. Temperature	The sender influences a third party node's hardware temperature using traffic of a flow. There must be a technique for the covert receiver to measure the temperature (indirectly).
PS1. Size Modulation	The covert channel uses the size of flow metadata (e.g. PDU size or size of a header element) to encode hidden messages.
PS2. Sequence Modulation	The covert channel alters the sequence of flow metadata to encode hidden information. (This pattern divides further into: P2.a. Position and P2.b. Number of Elements patterns.)
PS3. Add Redundancy	The covert channel embeds redundant metadata (e.g. by adding an unused IP option) in which data is hidden into a flow. Note that in comparison to PS1, the data is hidden in the redundant data's presence, not in the size of an PDU or header element.
PS10. Random Value	The covert channel embeds hidden data into flow metadata that contains a (pseudo-)random value.
PS11. Value Modulation	The covert channel selects one of the n values that a flow's metadata element can contain to encode a hidden message. (This pattern divides further into: P11.a. Case Pattern and P11.b. Least Significant Bit (LSB) patterns.)
PS12. Reserved/ Unused	The covert channel encodes hidden data into a flow's reserved or unused metadata elements.
PS20. Payload Field Size Modulation	The size of the payload in a flow is used to encode hidden information (this is a derivate of PS1 but for the payload since it involves the modification of a PDU's payload length field, i.e. PS1).
PS21. User data Corruption	The covert channel performs a (blind) insertion of covert data into a flow's payload (similar PT10).
PS30. Modify Redundancy	The covert channel compresses a flow's payload and the resulting free space is used to hide data.
PS31. User-data Value Modulation and Reserved/Unused	The covert channel performs a modification of a flow's payload in a way that is not reflected by PS30 and that does not result in a significantly modified interpretation of the data, e.g. by modifying least significant bits of digital images or hiding data in unused/reserved payload bits.

TABLE 9.2: Descriptions of hiding patterns in our extended taxonomy.

network covert channels are used simultaneously: one is used for the IPv4 protocol, the next for the TCP protocol, and finally the third is applied to HTTP. Pattern combination applies also to the case when, e.g. three separate connections are used for hidden data purposes and in each connection a separate network hiding pattern is utilized at the same time (e.g. IPv4-based in the first connection, TCP-based in the second, and HTTP-based in the last one). Typically such an approach is used in order to increase the overall steganographic bandwidth.

The second group of distributed covert channels realization is *pattern hopping* which allows to spread secret data in the temporal domain (time). In a nutshell it means that different patterns' utilization varies over time and thus they are applied sequentially for various (sub-)carriers. Usually, such an approach helps to improve the stealthiness of the covert data exchange as in order to detect it more "locations" must be monitored by the warden. An example of pattern hopping is the tool *PHCCT*. PHCCT implements a so-called *protocol hopping covert channel* that distributes data over different network protocols. To this end, PHCCT utilizes more than one pattern, namely *Add Redundancy* (embedded in HTTP) and *User-data Corruption* (embedded in FTP-Data).

Finally, the last group of techniques which allows to realize a distributed network covert channel is *pattern variation*. The original idea of pattern variation is that each of the defined patterns is considered in the certain context, i.e. the utilized hidden data carrier (e.g. a network protocol). In our case, we extend this view and define pattern variation in different contexts. In particular, three contexts can be distinguished: *host-based scattering*, *flow-based scattering*, and *protocol-based scattering* which will be described in detail with examples below. In all cases of pattern variation, the *same* pattern is applied to *different* contexts, i.e. its essence does not change.

Host-based scattering requires the covert sender and/or the covert receiver to control more than one physical host or other networking devices. Parts of the secret data are hidden in the legitimate traffic sent from or directed towards different hosts using the same pattern. An example of this kind of distributed covert channel is the SCTP multi-homing-based method (i.e. the host's ability to be visible in the network through more than one IP address) (Fraczek et al., 2012b). In such a scenario, each IP address of the covert receiver can be used to represent a single bit of secret data (or a sequence of bits). Then, by modulating the way that packets are addressed and sent secret data can be transferred in a distributed manner.

Next, *Flow-based* scattering takes advantage of the capability to set up multiple flows between two hosts and using them to signal secret data bits in a distributed way while utilizing the same pattern. This can be realized, for example, by dividing secret data into fragments and using a certain information hiding pattern (or several) to send each fragment using one of the available flows. An idea of using many flows for a distributed covert channel is exemplified by the Cloak method (Luo et al., 2007), which is a timing data hiding technique that encodes secret data bits by uniquely distributing N packets over M TCP flows. Please note that while in the case of *pattern hopping* a utilization of multiple flows is possible as well, *flow-based scattering* serves under the umbrella of pattern variation, i.e. it is required

to apply the *same pattern* to *different flows*, and pattern hopping must apply *different patterns*.

Finally, *Protocol-based* scattering applies a pattern to different communication protocols instead of hosts or flows. In contrast to flow-based scattering, it does not necessarily utilize flows of the same protocol but changes the actual protocol (which can generate multiple flows, too). This group is exemplified by *protocol switching covert channels* (PSCC), cf. Wendzel and Zander (2012). These channels assign hidden information to network protocols. For instance, one could link the HTTP protocol to the hidden value "0" and the DNS protocol to the hidden value "1". Then, by sending the packet sequence HTTP, DNS, DNS, HTTP, one would transfer the secret information "0110".

Obviously, there are other possibilities to create distributed network covert channels by developing mixed solutions so that it involves the parallel use of, e.g. pattern hopping and pattern variation or any other fusion of the concepts mentioned above.

9.5 Conclusion

We identified limitations of the existing pattern-based taxonomy, most importantly a lack of payload-based hiding patterns and a limited definition of distributed covert channels. For this reason, we extended the list of existing hiding patterns for network covert channels and their related taxonomy. We also extended the description of hybrid/distributed hiding methods and proposed an extension and improvement of the related concepts (especially pattern variation to handle multi-host, multi-flow and multi-protocol techniques). We hope this work will help to derive new insights into existing and new data hiding techniques.

Future work will be devoted to analyzing relationships between patterns with respect to their joint occurrence in existing methods as well as we will investigate whether any new data hiding methods can be deuced based on the less obvious pattern mixes.

Part V

Final Remarks

Chapter 10

Summary & Future Work

This thesis provided an extended scientific foundation for network information hiding. To this end, it contributed multiple novel taxonomies and improved the terminology of the domain, especially by the introduction of *hiding patterns*. Hiding patterns lower the chances for scientific re-inventions and support the unified understanding of how hiding methods work.

This thesis also proposed an improved review process for hiding techniques based on patterns. The review process considers the applicability and novelty (creativity) of proposed hiding techniques before deciding about their acceptance and future status. This altered review process was designed in a way to render it attractive for its users (reviewers) and it is another attempt to minimize scientific re-inventions and taxonomical inconsistencies.

Next, we introduced the first approach to unify the descriptions of hiding methods, rendering them ultimately comparable and therefore aiding the scientific process, progress, and understanding in the domain.

Moreover, this thesis conducted the first replication study in network steganography by performing original and extended experiments for a well-known countermeasure that was originally introduced by Cabuk et al., and that received 130+ citations so far. The results have shown that the extended results underline the limitations of the original approach in much more detail than originally published by the authors.

Additionally, the concept of *countermeasure variation* was proposed. Countermeasure variation is the modification of a hiding method so that it can detect covert channels of other hiding patterns than originally foreseen by the inventors of the countermeasure. We have experimentally verified that countermeasure variation can be applied to target different hiding patterns. Our experiments have shown that countermeasure variation can lead to high-quality detection results in terms of accuracy, precision, recall, and AUC. However, countermeasure variation was not able to provide high-quality results for *all* tested covert channel types and configurations.

Future work will involve performing additional countermeasure variations. Partially, these countermeasure variations are already in progress (e.g., in (Wendzel et al., 2019), we performed a countermeasure variation for the so-called *regularity* metric). Another challenge is to find approaches that render countermeasures dynamic and adaptive to limit side-effects on legitimate traffic while providing a higher performance. Also, it would be

beneficial for the whole research community to conduct experimental replication studies for countermeasures, e.g., under the umbrella of (intensively supervised bachelor or) master theses in computer science programs (or related degree programs). First theses-based replication studies are currently in progress under the supervision of the author.

In addition, novel approaches for active countermeasures, such as our recently introduced *dynamic warden* (Mazurczyk et al., 2019a), could lead to improved pattern-based covert channel limitation and elimination. Another aspect that can be considered in future work is to continue with the recently started analysis of reversibility for hiding patterns (Mazurczyk et al., 2019b) and to continue with the ongoing evaluation of pattern-based teaching in higher education (Wendzel and Mazurczyk, 2016). Finally, (network) steganography methods in cyber-physical systems were just recently described, cf. (Wendzel et al., 2017c; Wendzel et al., 2017a), and could be investigated on a per-pattern basis, too.

Appendix A

Publications and Awards

Publications of the Author

Edited Publications:

1. Steffen Wendzel, Wojciech Mazurczyk, Luca Caviglione, Amir Houmansadr (Eds.): "Emerging Topics in Defending Networked Systems", *Future Generation Computer Systems*, Elsevier, 2020 (**in preparation**).
2. Steffen Wendzel, Luca Caviglione, Alessandro Checco, Aleksandra Mileva, Jean-Francois Lalande, Wojciech Mazurczyk (Eds.): "Information Security Methodology, Replication Studies and Information Security Education", *Journal of Universal Computer Science*, University of Graz, 2020 (**in preparation**).
3. Zhihan Lv, Wojciech Mazurczyk, Steffen Wendzel, Houbing Song (Eds.): "Cyber-Physical Security in Industrial Environments", *IEEE Transactions on Industrial Informatics*, IEEE, 2019 (**in press**).
4. Steffen Wendzel, Luca Caviglione, Alessandro Checco, Aleksandra Mileva, Jean-Francois Lalande, Wojciech Mazurczyk (Eds.): "Proceedings of the First Int. Workshop on Information Security Methodology and Replication Studies", *Part of ARES 2019 Conference Proceedings*, ACM, 2019.
5. Wojciech Mazurczyk, Luca Caviglione, Steffen Wendzel (Eds.): "Recent Advancements in Digital Forensics, pt. 2", *IEEE Security & Privacy Magazine*, Vol. 17(1), IEEE, 2019.
6. Luca Caviglione, Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander (Eds.): "Emerging and Unconventional: New Attacks and Innovative Detection Techniques", *Security and Communication Networks (SCN)*, Wiley-Hindawi, 2018.
7. Wojciech Mazurczyk, Luca Caviglione, Steffen Wendzel (Eds.): "Recent Advancements in Digital Forensics, pt. 1", *IEEE Security & Privacy Magazine*, Vol. 15(6), IEEE, 2017.
8. Steffen Wendzel, Jörg Keller (Eds.): "Security, Privacy and Reliability of Smart Buildings", *Journal of Universal Computer Science (J.UCS)*, Vol. 22(9), Technische Universität Graz, 2016.

9. Michael Meier, Delphine Reinhardt, Steffen Wendzel (Eds.): "Tagungsband der Sicherheit 2016 – Beiträge der 8. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)", *LNI Vol. 256*, GI, 2016.
10. Christoph Pohl, Sebastian Schinzel, Steffen Wendzel (Eds.): "Proceedings of the Eight GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING)", *SR-2013-01*, Gesellschaft für Informatik (GI), 2013.

Textbooks:

1. Steffen Wendzel: "IT-Sicherheit für TCP/IP- und IoT-Netzwerke", 1st Ed., Springer, ISBN 978-3-658-22603-9, 2018.
2. Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, Krzysztof Szczypiorski: "Information Hiding in Communication Networks: Fundamentals, Mechanisms, and Applications", *IEEE Series on Information and Communication Networks Security*, 1st Ed., Wiley-IEEE, ISBN 978-1-118-86169-1, 2016.
3. Steffen Wendzel, Johannes Plötner: "Einstieg in Linux", 8th Ed., Rheinweg Verlag, ISBN 978-3-8362-6769-4, 2019, *7 older editions of this book are not listed here*.
4. Johannes Plötner, Steffen Wendzel: "Linux. Das umfassende Handbuch", 5th Ed., Galileo Press, ISBN 978-3-8362-1822-1, 2012, *4 older editions of this book are not listed here*.
5. Steffen Wendzel: "Tunnel und verdeckte Kanäle im Netz – Grundlagen, Protokolle, Sicherheit und Methoden", 1st Ed., Springer-Vieweg, ISBN 978-3-8348-1640-5, 2012.
6. Johannes Plötner, Steffen Wendzel: "Praxisbuch Netzwerksicherheit", 2nd Ed., Galileo Press, ISBN 978-3-89842-828-6, 2007, *1 older edition of this book is not listed here*.

Book Chapters:

1. Luca Caviglione, Wojciech Mazurczyk, Steffen Wendzel: "Advanced Information Hiding Techniques for Modern Botnets", in: Georgios Kambourakis, Marios Anagnostopoulos, Weizhi Meng and Peng Zhou (Eds.): *Botnets: Architectures, Countermeasures, and Challenges*, Ch. 4, pp. 165–188, CRC Press, 2019.
2. Steffen Wendzel, Detlef Olschewski: "Internet of Things und Smart Contracts: Risiken bei d. Digitalisierung v. Unternehmen", in: Thomas Barton, Christian Müller, Christian Seel (Eds.): *Angewandte Wirtschaftsinformatik*, Vol. III, Ch. 16, pp. 291–302, Springer, 2018.
3. Steffen Wendzel, Jernej Tonejc, Jaspreet Kaur, Alexandra Kobekova: "Security of Smart Buildings", in: Houbing Song, Glenn A. Fink, Sabina Jeschke (Eds.): *Security and Privacy in Cyber-Physical Systems: Foundations and Applications*, Ch. 16, pp. 327-352, Wiley-IEEE, 2017.
4. Steffen Wendzel, Jörg Keller: "Einführung in die Forschungsthematik der verdeckten Kanäle", in: Jörg Sambleben, Stefan Schumacher (Eds.): *Informationstechnologie und Sicherheitspolitik*, pp. 91-102, BoD, 2012.

Journal Articles:

1. Aleksandar Velinov, Aleksandra Mileva, Steffen Wendzel, Wojciech Mazurczyk: "Covert Channels in the MQTT-based Internet of Things", in: *IEEE Access*, IEEE, **in press**.
2. Steffen Wendzel, Cédric Lévy-Bencheton, Luca Caviglione: "Not all Areas are Equal: Analysis of Citations in Information Security Research", in: *Scientometrics*, Springer, **in press**.
3. Steffen Wendzel, Florian Link, Daniela Eller, Wojciech Mazurczyk: "Detection of Size Modulation Covert Channels Using Countermeasure Variation", in: *Journal of Universal Computer Science (J.UCS)*, Technische Universität Graz, **in press**.
4. Wojciech Mazurczyk, Steffen Wendzel, Mehdi Chourib, Jörg Keller: "Countering Adaptive Network Covert Communication with Dynamic Wardens", in: *Future Generation Computer Systems*, Vol. 94, pp. 712-725, Elsevier, 2019.
5. Ralf Keidel, Steffen Wendzel, Sebastian Zillien, Eric S. Conner, Georg Haas: "WoDiCoF – A Testbed for the Evaluation of (Parallel) Covert Channel Detection Algorithms", in: *Journal of Universal Computer Science*, Vol. 24(5), pp. 556-576, Technische Universität Graz, 2018.
6. Krzysztof Cabaj, Luca Caviglione, Wojciech Mazurczyk, Steffen Wendzel, Alan Woodward, Sebastian Zander: "The New Threats of Information Hiding: the Road Ahead", in: *IEEE IT Professional*, Vol. 20(3), pp. 31-39, IEEE, 2018.
7. Wojciech Mazurczyk, Steffen Wendzel: "Information Hiding: Challenges for Forensic Experts", in: *Communications of the ACM*, Vol. 61(1), pp. 86-94, ACM, 2018.
8. Luca Caviglione, Steffen Wendzel, Wojciech Mazurczyk: "The Future of Digital Forensics: Challenges and the Road Ahead", in: *IEEE Security & Privacy Magazine*, Vol. 15(6), pp 12-17, IEEE, 2017.
9. Steffen Wendzel, Luca Caviglione, Wojciech Mazurczyk, Jean-Francois Lalande: "Network Information Hiding and Science 2.0: Can it be a Match?", in: *International Journal of Electronics and Telecommunications*, Vol. 63(2), pp. 217-222, 2017.
10. Steffen Wendzel, Wojciech Mazurczyk, Georg Haas: "Steganography for Cyber-physical Systems", in: *Journal of Cyber Security and Mobility (JCSM)*, Vol. 6(2), pp. 105-126, River Publishers, 2017.
11. Gabriele Spenger, Jörg Keller, Steffen Wendzel: "Enhanced Ant Colony-inspired Parallel Algorithm to Improve Cryptographic PRNGs", in: *Journal of Cyber Security and Mobility (JCSM)*, Vol. 6(2), pp. 147-170, River Publishers, 2017.
12. Steffen Wendzel: "How to Increase the Security of Smart Buildings", in: *Communications of the ACM (CACM)*, Vol. 59(5), ACM, 2016.
13. Steffen Wendzel, Wojciech Mazurczyk, Sebastian Zander: "Unified Description for Network Information Hiding Methods", in: *Journal*

of Universal Computer Science (J.UCS), Vol. 22(11), pp. 1456-1486, Technische Universität Graz, 2016.

14. Steffen Wendzel, Saffija Kasem-Madani: "IoT Security: The Improvement-Decelerating 'Cycle of Blame"', in: *Journal of Cyber Security and Mobility (JCSM)*, River Publishers, DOI: 10.13052/popcas010, 2016.
15. Jaspreet Kaur, Steffen Wendzel, Omar Eissa, Jernej Tonejc, Michael Meier: "Covert Channel-internal Control Protocols: Attacks and Defense", in: *Security and Communication Networks (SCN)*, Vol. 9(15), pp. 2986–2997, John Wiley & Sons, 2016.
16. Matthias Naumann, Steffen Wendzel, Wojciech Mazurczyk, Jörg Keller: "Micro Protocol Engineering for Unstructurred Carriers", in: *Security and Communication Networks (SCN)*, Vol. 9(15), pp. 2972-2985, John Wiley & Sons, 2016.
17. Wojciech Mazurczyk, Steffen Wendzel, Ignacio Azagra Villares, Krzczystof Szczypiorski: "On Importance of Steganographic Cost for Network Steganography", in: *Security and Communication Networks (SCN)*, Vol. 9(8), John Wiley & Sons, 2016.
18. Steffen Wendzel, Sebastian Zander, Bernhard Fechner, Christian Herdin: "A Pattern-based Survey and Categorization of Network Covert Channel Techniques", in: *ACM Computing Surveys (CSUR)*, Vol. 47(3), ACM, 2015.
19. Steffen Wendzel, Carolin Palmer: "Creativity in Mind: Evaluating and Maintaining Advances in Network Steganographic Research", in: *Journal of Universal Computer Science (J.UCS)*, Vol. 21(12), Technische Universität Graz, 2015.
20. Krzysztof Szczypiorski, Artur Janicki, Steffen Wendzel: "'The Good, The Bad And The Ugly': Evaluation of Wi-Fi Steganography", in: *Journal of Communications (JCM)*, Vol. 10(10), 2015.
21. Steffen Wendzel, Jörg Keller: "Hidden and Under Control – A Survey and Outlook on Covert Channel-internal Control Protocols", in: *Springer Annals of Telecomunication (ANTE)*, Vol. 69, no. 7-8, Springer Paris, 2014.
22. Steffen Wendzel, Jörg Keller: "IT-gestütztes Management und Controlling: Verdeckte Kanäle – eine zunehmende Gefahr für Unternehmensdaten", in: *Controlling (Zeitschrift für erfolgsorientierte Unternehmenssteuerung)*, Vol. 14(6)/26. Jahrgang, Beck, 2014.
23. Steffen Wendzel, Jörg Keller: "Preventing Protocol Switching Covert Channels", in: *International Journal on Advances in Security*, Vol. 5(3&4), Iaria, 2012.

Conference and Workshop Publications:

1. Steffen Wendzel: "Protocol-independent Detection of "Messaging Ordering" Network Covert Channels", in Proc. *14th International Conference on Availability, Reliability and Security, ARES (CUING Workshop)*, pp. 63:1-63:8, ACM, 2019.

2. Wojciech Mazurczyk, Przemysław Szary, Steffen Wendzel, Luca Caviglione: "Towards Reversible Storage Network Covert Channels", in Proc. *14th International Conference on Availability, Reliability and Security, ARES (CUING Workshop)*, pp. 69:1-69:8, ACM, 2019.
3. Tobias Schmidbauer, Steffen Wendzel, Aleksandra Mileva, Wojciech Mazurczyk: "Introducing Dead Drops to Network Steganography using ARP-Caches and SNMP-Walks", in Proc. *14th International Conference on Availability, Reliability and Security, ARES (CUING Workshop)*, pp. 64:1-64:10, ACM, 2019.
4. Sebastian Zillien, Steffen Wendzel: "Detection of covert channels in TCP retransmissions", in Proc. *23rd Nordic Conference on Secure IT Systems (NordSec)*, LNCS Vol. 11252, pp. 203-218, Springer, 2018.
5. Steffen Wendzel, Daniela Eller, Wojciech Mazurczyk: "One Countermeasure, Multiple Patterns: Countermeasure Variation for Covert Channels", in Proc. *Central European Security Conference (CECC'18)*, pp. 1:1–1:6, ACM, 2018.
6. Wojciech Mazurczyk, Steffen Wendzel, Krzysztof Cabaj: "Towards Deriving Insights into Data Hiding Methods Using Pattern-based Approach", in Proc. *13th International Conference on Availability, Reliability and Security, ARES (CUING Workshop)*, pp. 10:1-10:10, ACM, 2018.
7. Steffen Wendzel: "Get Me Cited, Scotty! Analysis of Citations in Covert Channel/Steganography Research", in Proc. *13th International Conference on Availability, Reliability and Security, ARES (CUING Workshop)*, pp. 13:1-13:8, ACM, 2018.
8. Florian Lehner, Wojciech Mazurczyk, Jörg Keller, Steffen Wendzel: "Inter-protocol Steganography for Real-time Services and Its Detection Using Traffic Coloring Approach", in Proc. *42th IEEE Conference on Local Computer Networks (LCN)*, pp. 78-85, IEEE, 2017.
9. Steffen Wendzel, Wojciech Mazurczyk, Georg Haas: "Don't You Touch My Nuts: Information Hiding In Cyber Physical Systems Using Smart Buildings", in Proc. *IEEE Security and Privacy Workshops (SPW)*, pp. 29-34, IEEE, 2017.
10. Jörg Keller, Gabriele Spenger, Steffen Wendzel: "Ant Colony-inspired Parallel Algorithm to Improve Cryptographic Pseudo Random Number Generators", in Proc. *IEEE Security and Privacy Workshops (SPW)*, pp. 12-22, IEEE, 2017.
11. Steffen Wendzel, Wojciech Mazurczyk: "An Educational Network Protocol for Covert Channel Analysis Using Patterns", in Proc. *23rd Conference on Computer and Communications Security (CCS)*, pp. 1739-1741, ACM, 2016.
12. Jaspreet Kaur, Jernej Tonejc, Steffen Wendzel, Michael Meier: "Securing BACnet's Pitfalls", in Proc. *30th IFIP TC 11 International Conference on ICT Systems Security and Privacy Protection (SEC 2015)*, IFIP AICT 455, pp. 616-629, Springer, 2015.

13. Eva Maria Anhaus, Steffen Wendzel: "BACtag – Data Leakage Protection für Gebäude", in Proc. *D-A-CH Security 2015*, pp. 417-248, syssec, 2015.

14. Jernej Tonejc, Jaspreet Kaur, Adrian Karsten, Steffen Wendzel: "Visualizing BACnet Data to Facilitate Humans in Building-Security Decision-Making", in Proc. *HCII Conference for Human Aspects of Information Security, Privacy and Trust (HAS)*, LNCS 9190, pp. 693-704, Springer, 2015.

15. Jaspreet Kaur, Steffen Wendzel, Michael Meier: "Countermeasures for Covert Channel-internal Control Protocols", in Proc. *10th International Conference on Availability, Reliability and Security, ARES (IWCC Workshop)*, pp. 422-428, IEEE, 2015.
Paper received a **best paper award**.

16. Luca Caviglione, Jean-Francois Lalande, Wojciech Mazurczyk, Steffen Wendzel: "Analysis of Human Awareness of Security and Privacy Threats in Smart Environments", in Proc. *HCII Conference for Human Aspects of Information Security, Privacy and Trust (HAS)*, LNCS 9190, pp. 165-177, Springer, 2015.

17. Jaspreet Kaur, Christian Herdin, Jernej Tonejc, Steffen Wendzel, Michael Meier, Sebastian Szlósarczyk: "Novel Approaches for Security in Building Automation Systems", in Proc. *14. Deutscher IT-Sicherheitskongress des BSI*, pp. 148-155, 2015.

18. Steffen Wendzel, Wojciech Mazurczyk, Luca Caviglione, Michael Meier: "Hidden and Uncontrolled – On the Emergence of Network Steganography", in Proc. *Information Security Solutions Europe (ISSE)*, pp. 123-133, Springer-Vieweg, 2014.

19. Steffen Wendzel, Christian Herdin, Roman Wirth, Masood Masoodian, Saturnino Luz, Jaspreet Kaur: "Mosaic Chart-based Visualization in Building Automation Systems", in Proc. *9th Security Research Conference „Future Security"*, pp. 687-690, Fraunhofer Verlag, 2014.

20. Steffen Wendzel, Viviane Zwanger, Michael Meier, Sebastian Szlósarczyk: "Envisioning Smart Building Botnets", in Proc. *GI Sicherheit 2014*, LNI 228, pp. 319-329, GI, 2014.

21. Sebastian Szlósarczyk, Steffen Wendzel, Jaspreet Kaur, Michael Meier, Frank Schubert: "Towards Suppressing Attacks on and Improving Resilience of Building Automation Systems – An Approach Exemplified Using BACnet", in Proc. *GI Sicherheit 2014*, LNI 228, pp. 407-418, GI, 2014.

22. Jaspreet Kaur, Steffen Wendzel: "Realization and Experiences with a Low-Cost Building Automation Security Testbed for Educational Purpose", in Proc. *Computer Science Conference for University of Bonn Students (CSCUBS)*, University of Bonn, 2014.

23. Jean-Francois Lalande, Steffen Wendzel: "Hiding Privacy Leaks in Android Applications Using Low-Attention Raising Covert Channels", in Proc. *International Conference on Availability, Reliability and Security,*

ARES (International Workshop on Emerging Cyberthreats and Countermeasures), pp. 701-710, IEEE, 2013.

24. Benjamin Kahler, Steffen Wendzel: "How to own a Building? Wardriving gegen die Gebäude-Automation", in Proc. *20. DFN Workshop zur Sicherheit in vernetzten Systemen*, pp. H1-H13, BoD, 2013.

25. Peter Backs, Steffen Wendzel, Jörg Keller: "Dynamic Routing in Covert Channel Overlays Based on Control Protocols", in Proc. *International Workshop on Information Security, Theory and Practice (ISTP-2012)*, pp. 32-39, IEEE, 2012.

26. Steffen Wendzel, Benjamin Kahler, Thomas Rist: "Covert Channels and their Prevention in Building Automation Protocols – A Prototype Exemplified Using BACnet", in Proc. *IEEE GreenCOM/CPSCom/iThings (2nd Workshop on Security of Systems and Software Resiliency)*, pp. 731-736, IEEE, 2012.

27. Steffen Wendzel, Sebastian Zander: "Detecting Protocol Switching Covert Channels", in Proc. *37th IEEE Conference on Local Computer Networks (LCN)*, pp. 280-283, IEEE, 2012.

28. Steffen Wendzel, Jörg Keller: "Systematic Engineering of Control Protocols for Covert Channels", in Proc. *13th Joint IFIP TC6 and TC11 Conference on Communications and Multimedia Security (CMS 2012)*, LNCS 7394, pp. 131-144, Springer, 2012.

29. Steffen Wendzel: "Covert and Side Channels in Buildings and the Prototype of a Building-aware Active Warden", in Proc. *IEEE International Workshop on Security and Forensics in Communication Systems (SFCS 2012)*, pp. 8339-8344, IEEE, 2012.

30. Steffen Wendzel, Jörg Keller: "Design and Implementation of an Active Warden Addressing Protocol Switching Covert Channels", in Proc. *7th International Conference on Internet Monitoring and Protection (ICIMP 2012)*, pp. 1-6, Iaria, 2012.
Paper received a **best paper award**.

31. Thomas Rist, Steffen Wendzel, Masood Masoodian, Elisabeth André: "Next-Generation Home Automation Systems", in Proc. *Techniken für Menschen im nächsten Jahrzehnt – Beiträge zum Usability Day X*, pp. 80-87, Pabst Science Publishers, 2012.

32. Steffen Wendzel: "The Problem of Traffic Normalization Within a Covert Channel's Network Environment Learning Phase", in Proc. *GI Sicherheit 2012*, LNI Vol. 195, pp. 149-161, Gesellschaft für Informatik (GI), 2012.

33. Steffen Wendzel, Jörg Keller: "Low-attention forwarding for mobile network covert channels", in Proc. *12th Conference on Communications and Multimedia Security (CMS 2011)*, LNCS Vol. 7025, pp. 122-133, Springer, 2011.

34. Steffen Wendzel, Thomas Rist, Elisabeth André, Masood Masoodian: "A secure interoperable architecture for building-automation applications", in Proc. *4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, pp. 8:1–8:5, ACM, 2011.
35. Thomas Rist, Steffen Wendzel, Masood Masoodian, Paul Monigatti, Elisabeth André: "Creating Awareness for Efficient Energy Use in Smart Homes", in Proc. *Intelligent Wohnen – Zusammenfassung der Beiträge zum Usability Day IX*, pp. 162-168, Pabst Science Publishers, 2011.

Theses:

1. Steffen Wendzel: "Novel Approaches for Network Covert Storage Channels", *Doctoral Dissertation*, Department of Mathematics & Computer Science, University of Hagen, 2013.

Contributions to EU Research Agenda Documents:

The author contributed sections, chapters or content to the following EU documents:

1. Pascal Bisson et al. (Eds.): "Security and Resilience of Smart Home Environments: Good Practices and Recommendations", *ENISA*, 2015.
2. Mari Kert et al. (Eds.): "State-of-the-art of Secure ICT Landscape, version 2", *EU NIS Platform Working Group III*, 2015.
3. Cédric Lévy-Bencheton et al. (Eds.): "Cybersecurity Strategic Research Agenda", *EU NIS Platform Working Group III*, 2015.

Other Publications:

1. Steffen Wendzel: Review of the article "Man-in-the-middle attacks on Secure Simple Pairing in Bluetooth Standard V5.0 and Its Countermeasure", ACM Computing Reviews Vol. 59, ACM, 2018.
2. Steffen Wendzel: Review of the article "Out-of-band Covert Channels– A Survey'', ACM Computing Reviews, Vol. 58, ACM, 2017.
3. Steffen Wendzel: Review of the Cambridge Univ. Press book "Darkweb Cyber Threat Intelligence Mining", ACM Computing Reviews, Vol. 58, 2017.
4. Steffen Wendzel: "Why Johnny Can't Use Stego: A Human-oriented Perspective on the Application of Steganography", *CoRR abs/1609.06664*, 2016.
5. Wojciech Mazurczyk, Philipp Amann, Luca Caviglione, Steffen Wendzel: "CUING: Criminal Use of Information Hiding Initiative", *European CIIP Newsletter*, Issue 25 (Vol. 10, No. 3), pp. 31-32, 2016.

Appendix B

Content of the Attached CD

The attached CD contains this habilitation thesis' PDF version to allow the reader an easier search and analysis of its content.

Appendix C

Short Biography of the Author

Steffen Wendzel received his Dipl.-Inf. (FH) degree in computer science from the Faculty of Computer Science of the Kempten University of Applied Sciences in 2009, his M.Sc. degree in computer science from the Faculty of Computer Science of the Augsburg University of Applied Sciences in 2011, and his Ph.D. degree (Dr. rer. nat.) from the Faculty of Mathematics and Computer Science of the University of Hagen in 2013.

From 2010 till 2013, he worked as a research associate at the Augsburg University of Applied Sciences; during that time, he spent several months at the HCI laboratory at the University of Waikato, New Zealand. He joined Frauhofer FKIE in 2013 and built up a team on smart building security research till 2016, when he joined Worms University of Applied Sciences as a professor for cyber security & computer networks. Since 2017 he is deputy scientific head of the *Zentrum für Technologie und Transfer* (ZTT) and head of the Cyber Security Research Group (CSRG) in Worms.

He published more than 140 works, including six books, of which several appeared in high-ranked journals (e.g. ACM Computing Surveys, Elsevier Future Generation Computer Systems, IEEE ACCESS, IEEE Security & Privacy, Scientometrics, and Communications of the ACM) and events (e.g. IEEE LCN, ACM CCS, and IFIP SEC). Steffen Wendzel organized special issues for several journals, including IEEE Transactions on Industrial Informatics (IF: 7.377) and Elsevier Future Generation Computer Systems (IF: 5.768). He is the initiator of the International Workshop on Information Security Methodology and Replication Studies (IWSMR) that was co-located with ARES 2019 and serves as an editorial board member for the Journal of Universal Computer Science and the Journal of Cyber Security & Mobility.

The author is, moreover, a member of the *Criminal Use of Information Hiding* initiative's steering committee, elected Member of EURASIP Special Area Team (SAT) on Biometrics, Data Forensics, and Security (BForSec) and was previously a leader of an *Area of Interest* for the EU NIS Public-Private Platform Working Group on Secure ICT research and innovation as well as member of the streeing committee of the *Arbeitskreis kritische Informations- und Kommunikationsinfrastrukturen* (AK KRITIS) of the German Informatics Society (GI) SIG Management für Informationssicherheit (SECMGT).

Bibliography

Agars, M. D., J. C. Kaufman, A. Deane, and B. Smith (2012). "Fostering Individual Creativity Through Organizational Context". In: *Handbook of organizational creativity*. Ed. by M. D. Mumford. New York, NY: Elsevier, pp. 271–291.

Ahsan, K. and D. Kundur (2002a). "Practical Data Hiding in TCP/IP". In: *Proc. Workshop on Multimedia Security at ACM Multimedia '02*.

Ahsan, Kamran and Deepa Kundur (2002b). "Practical data hiding in TCP/IP". In: *Proc. Workshop on Multimedia Security at ACM Multimedia*. Vol. 2(7). ACM.

Alavi, M. and D. Leidner (2001). "Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues". In: *MIS Quarterly* 25.25, pp. 107–136.

Alexander, C., S. Ishikawa, and M. Silverstein (1977). *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press.

Amabile, T.M. (1983). *The social psychology of creativity*. New York: Springer.

Anderson, Ross (1996). "Stretching the limits of steganography". In: *Proc. Information Hiding: First International Workshop Cambridge, U.K.* Ed. by Ross Anderson. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 39–48. ISBN: 978-3-540-49589-5. DOI: 10.1007/3-540-61996-8_30. URL: https://doi.org/10.1007/3-540-61996-8_30.

Anderson, Ross J. and Fabien Petitcolas (1998). "On the limits of steganography". In: *IEEE Journal on Selected Areas in Communications* 16.4, pp. 474–481. DOI: 10.1109/49.668971.

Anderson, Thomas (2009). "Conference reviewing considered harmful". In: *ACM SIGOPS Operating Systems Review* 43.2, pp. 108–116. ISSN: 01635980. DOI: 10.1145/1531793.1531815.

Authorea (2019). *Authorea*. URL: https://www.authorea.com.

Avi2ienis, A., J.-C. Laprie, B. Randell, and C. Landwehr (2004). "Basic Concepts and Taxonomy of Depandable and Secure Computing". In: *IEEE Trans. Depandable and Secure Computing* 1.1, pp. 11–33. DOI: 10.1109/TDSC.2004.2.

Backs, Peter, Steffen Wendzel, and Jörg Keller (2012). "Dynamic Routing in Covert Channel Overlays Based on Control Protocols". In: *Proc. International Workshop on Information Security, Theory and Practice (ISTP-2012)*. IEEE, pp. 32–39.

Bajpai, Vaibhav, Anna Brunstrom, Anja Feldmann, et al. (2019). "The Dagstuhl Beginners Guide to Reproducibility for Experimental Networking Research". In: *ACM SIGCOMM Computer Communication Review* 49.1. Editorial note, pp. 24–30.

Baughman, W. A. and M. D. Mumford (1995). "Process-analytic models of creative capacities: Operations influencing the combination-and-re- organization process". In: *Creativity Research Journal* 8.1, pp. 37–62.

Bender, W., D. Gruhl, N. Morimoto, and A. Lu (1996a). "Techniques for data hiding". In: *IBM Systems Journal* 35.3.4, pp. 313–336. ISSN: 0018-8670. DOI: 10.1147/sj.353.0313.

– (1996b). "Techniques for data hiding". In: *IBM Systems Journal* 35.3.4, pp. 313–336. ISSN: 0018-8670. DOI: 10.1147/sj.353.0313.

Benzel, T., R. Braden, D. Kim, et al. (2007). "Design, Deployment, and Use of the DETER Testbed". In: *Proc. DETER Community Workshop on Cyber-Security and Test*. USENIX Assoc.

Berk, V., A. Giani, and G. Cybenko (2005). *Detection of Covert Channel Encoding in Network Packet Delays*. Tech. rep. TR2005-536. Department of Computer Science, Dartmouth College. URL: http://www.ists.dartmouth.edu/library/149.pdf.

Bertolotti, F., E. Mattarelli, M. Vignoli, and D. M. Macrí (2015). "Exploring the relationship between multiple team membership and team performance: The role of social networks and collaborative technology". In: *Research Policy* 44, pp. 911–924. DOI: 10.1016/j.respol.2015.01.019.

Borland, T. (2008). *Guide to Encrypted Dynamic Covert Channels*. URL: http://turboborland.blogspot.com/2008/12/guide-to-encrypted-dynamic-covert.html.

Brinkley, Donald L and Roger R Schell (1995). "Concepts and terminology for computer security". In: *Information security: An integrated collection of essays*, pp. 40–97.

Buchanan, W. J. and D. Llamas (2004). "Covert channel analysis and detection with reverse proxy servers using Microsoft Windows". In: *Proc. 3rd European conference on information warfare and security*, pp. 31–40.

Bücheler, Thierry and Jan Henrik Sieg (2011). "Understanding Science 2.0: Crowdsourcing and Open Innovation in the Scientific Method". In: *Procedia Computer Science* 7.0, pp. 327 –329. ISSN: 1877-0509. DOI: 10.1016/j.procs.2011.09.014.

Cabaj, Krzysztof, Luca Caviglione, Wojciech Mazurczyk, Steffen Wendzel, Alan Woodward, and Sebastian Zander (2018a). "The New Threats of Information Hiding: the Road Ahead". In: *IEEE IT Professional* 20.3, pp. 31–39. DOI: 10.1109/MITP.2018.032501746.

Cabaj, Krzysztof, Wojciech Mazurczyk, Piotr Nowakowski, and Piotr Żórawski (2018b). "Towards Distributed Network Covert Channels Detection Using Data Mining-based Approach". In: *Proceedings of the 13th International Conference on Availability, Reliability and Security*. ARES 2018. Hamburg, Germany: ACM, 12:1–12:10. ISBN: 978-1-4503-6448-5. DOI: 10.1145/3230833.3233264. URL: http://doi.acm.org/10.1145/3230833.3233264.

Cabuk, S. (2006). "Network covert channels: Design, analysis, detection, and elimination". PhD thesis. Purdue University.

Cabuk, S., C. E. Brodley, and C. Shields (2009). "IP Covert Channel Detection". In: *ACM Transactions on Information and System Security (TISSEC)* 12.4, 22:1–22:29.

Cabuk, Serdar, Carla E. Brodley, and Clay Shields (2004). "IP covert timing channels: design and detection". In: *Proceedings of the 11th ACM conference on Computer and communications security*. CCS '04. Washington DC, USA: ACM, pp. 178–187. ISBN: 1-58113-961-6. DOI: 10.1145/1030083.1030108. URL: http://doi.acm.org/10.1145/1030083.1030108.

Cai, Xiang, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson (2012). "Touching from a distance: Website fingerprinting attacks and defenses". In: *Proceedings of the 2012 ACM conference on Computer and Communications Security*. ACM, pp. 605–616.

Calhoun Jr, Telvis E., Xiaojun Cao, Yingshu Li, and Raheem Beyah (2012). "An 802.11 MAC layer covert channel". In: *Wireless Communication Mobile Computing* 12, pp. 393–405.

Caroff, X. and T. Lubart (2012). "Multidimensional Approach to Detecting Creative Potential in Managers". In: *Creativity Research Journal* 24.11, pp. 13–20.

Carrara, B. and C. Adams (2016). "Out-of-Band Covert Channels – A Survey". In: *ACM Computing Surveys* 49.2, 23:1–36.

Chen, Olga, Catherine Meadows, and Gautam Trivedi (2017). "Stealthy Protocols: Metrics and Open Problems". In: *Concurrency, Security, and Puzzles: Essays Dedicated to Andrew William Roscoe on the Occasion of His 60th Birthday*. Cham: Springer International Publishing, pp. 1–17. ISBN: 978-3-319-51046-0. DOI: 10.1007/978-3-319-51046-0_1. URL: https://doi.org/10.1007/978-3-319-51046-0_1.

Cornell University (2015). *ArXiv Website*. URL: http://arxiv.org.

Craver, S. (1998). "On Public-Key Steganography in the Presence of an Active Warden". In: *Proc. Information Hiding*. Vol. 1525. LNCS. Springer, pp. 355–368. ISBN: 978-3-540-65386-8.

Csikszentmihaly, M. (1996). *Creativity: flow and the psychology of discovery and invention*. New York: Harper Collins.

daemon9 (1997). "LOKI2 (the implementation)". In: *Phrack Magazine* 7.51. URL: http://www.phrack.org/issues.html?issue=51&id=6.

Danezis, George and Claudia Diaz (2008). *A Survey of Anonymous Communication Channels*. Tech. rep. MSR-TR-2008-35. Microsoft Research.

Daniel E. Geer, Jr. (2019). "Unknowable Unknowns". In: *IEEE Security and Privacy* 17.2, pp. 79–80.

Dean, Jeffrey and Sanjay Ghemawat (2004). "MapReduce: Simplified Data Processing on Large Clusters". In: *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*. OSDI'04. San Francisco, CA: USENIX Association, pp. 10–10. URL: http://dl.acm.org/citation.cfm?id=1251254.1251264.

deGraaf, Rennie, John Aycock, and Michael Jacobson Jr. (2005). "Improved Port Knocking with Strong Authentication". In: *Proceedings of 21st Annual Computer Security Applications Conference*.

Department of Defense (1985). *Trusted Computer System Evaluation Criteria (TCSEC, DoD 5200.28-STD, Orange Book)*.

Dingledine, R., N. Mathewson, and P. Syverson (2004). "Tor: The Second-generation Onion Router". In: *USENIX Security*.

Donaldson, A. L., J. McHugh, and K. A. Nyberg (1988). "Covert Channels in Trusted LANs". In: *Proc. of 11th NBS/NCSC National Computer Security Conference*, pp. 226–232.

Drazin, R., M.A. Glynn, and R.K. Kazanjian (1999). "Multilevel theorizing about creativity in organizations: A sensemaking perspective". In: *Academy of Management Review* 24, pp. 286–307.

Dyatlov, A. and S. Castro (2005). *Exploitation of data streams authorized by a network access control system for arbitrary data transfers: tunneling and covert channels over the HTTP protocol*. Tech. rep. Gray-World.net.

Dyer, Kevin P, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton (2012). "Peek-a-Boo, I still see you: Why efficient traffic analysis countermeasures fail". In: *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, pp. 332–346.

El-Atawy, A. and E. Al-Shaer (2009). "Building Covert Channels over the Packet Reordering Phenomenon". In: *INFOCOM 2009*, pp. 2186–2194.

Elsadig, Muawia A. and Yahia A. Fadlalla (2017). "A Balanced Approach to Eliminate Packet Length-based Covert Channels". In: *Proc. 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS)*. IEEE, pp. 1–7.

Engel, J., C. Märtin, and P. Forbrig (2011). "HCI Patterns as a Means to Transform Interactive User Interfaces to Diverse Contexts of Use". In: *Proc. Human-Computer Interaction. Design and Development Approaches*. Vol. 6761. LNCS. Springer Berlin Heidelberg, pp. 204–213.

Engel, J., C. Märtin, C. Herdin, and P. Forbig (2013). "Formal Pattern Specifications to Facilitate Semi-Automated User Interface Generation". In: *Proc. 15th HCI International*. Vol. 8004. LNCS. Springer, pp. 300–309.

Esser, H.-G. (2005). "Ausnutzung verdeckter Kanaele am Beispiel eines Web-Servers". (in German). MA thesis. RWTH Aachen.

Farmer, S.M., P. Tierney, and K. Kung-Mcintyre (2003). "Employee creativity in Taiwan: an application of role identity theory". In: *Academy of Management Journal* 46, pp. 618–630.

Fincher, S. (2003). "Perspectives on HCI patterns: concepts and tools (introducing PLML)". In: *Interfaces* 56, pp. 182–196.

Fincher, S., J. Finlay, S. Greene, et al. (2003). "Perspectives on HCI patterns: concepts and tools". In: *CHI'03 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '03. Ft. Lauderdale, Florida, USA: ACM, pp. 1044–1045.

Fisk, G., M. Fisk, C. Papadopoulos, and J. Neil (2003). "Eliminating Steganography in Internet Traffic with Active Wardens". In: *Revised Papers from the 5th International Workshop on Information Hiding*. Springer, pp. 18–35.

Fraczek, W., W. Mazurczyk, and K. Szczypiorski (2012a). "Multi-level Steganography: Improving Hidden Communication in Networks". In: *Journal of Universal Computer Science (J. UCS)* 18.14, pp. 1967–1986.

Fraczek, Wojciech, Wojciech Mazurczyk, and Krzysztof Szczypiorski (2012b). "Hiding Information in a Stream Control Transmission Protocol". In: *Comput. Commun.* 35.2, pp. 159–169. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2011.08.009. URL: http://dx.doi.org/10.1016/j.comcom.2011.08.009.

Franzen, Martina (2018). "Die digitale Transformation der Wissenschaft". In: *Beiträge zur Hochschulforschung* 40.4. in German, pp. 8–28.

Franzoni, Chiara and Henry Sauermann (2014). "Crowd science: The organization of scientific research in open collaborative projects". In: *Research Policy* 43.1, pp. 1 –20. ISSN: 0048-7333. DOI: 10.1016/j.respol.2013.07.005.

Freiling, Felix, Rüdiger Grimm, Karl-Erwin Großpietsch, et al. (2014). "Technische Sicherheit und Informationssicherheit". In: *Informatik-Spektrum* 37.1. (in German), pp. 14–24.

Fridrich, J. (1999). "Applications of data hiding in digital images". In: *Signal Processing and Its Applications, 1999. ISSPA '99. Proceedings of the Fifth International Symposium on*. Vol. 1, 9 vol.1–. DOI: 10.1109/ISSPA.1999.818099.

Fridrich, Jessica (2009). *Steganography in Digital Media – Principles, Algorithms, and Applications*. New York, NY: Cambridge University Press.

Gaffar, A., D. Sinnig, A. Seffah, and P. Forbrig (2004). "Modeling patterns for task models". In: *Proceedings of the 3rd annual Conference on Task Models and Diagrams*. TAMODIA '04. Prague, Czech Republic: ACM, pp. 99–104.

Gamma, E., R. Helm, R. Johnson, and J. Vlissides (1995). *Design patterns: elements of reusable object-oriented software*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0-201-63361-2.

Garcia, Luis Martin and Fyodor (2017). *Nping*. URL: https://nmap.org/nping/.

Geddes, John, Max Schuchard, and Nicholas Hopper (2013). "Cover your ACKs: pitfalls of covert channel censorship circumvention". In: *Proceedings of the 2013 ACM SIGSAC conference on Computer & Communications Security (CCS)*. ACM, pp. 361–372.

Gentner, D., S. Brem, R. Ferguson, P. Wolff, A. B. Markman, and K. Forbus (1997). "Analogy and creativity in the works of Johannes Kepler". In: *Creative thought: An investigation of conceptual structures and processes*. Ed. by T. B. Ward, S. M. Smith, and J. Vaid. Washington D.C.: American Psychological Association, pp. 403–459.

Getchell, A. (2008). *RE: For those interested in covert channels*. A posting on the securityfocus penetration testing mailinglist. URL: http://www.securityfocus.com/archive/101/499640.

Gianvecchio, S. and H. Wang (2007). "Detecting Covert Timing Channels: An Entropy-Based Approach". In: *Proceedings of 14th ACM Conference on Computer and Communication Security (CCS)*. Alexandria, VA, USA.

Gianvecchio, Steven, Haining Wang, Duminda Wijesekera, and Sushil Jajodia (2008). "Model-Based Covert Timing Channels: Automated Modeling and Evasion". In: *Proceedings of Recent Advances in Intrusion Detection (RAID) Symposium*. URL: http://www.cs.wm.edu/~srgian/paper/raid08.pdf.

Giffin, J., R. Greenstadt, P. Litwack, and R. Tibbetts (2003). "Covert messaging through TCP timestamps". In: *Proc. 2nd International Conference on Privacy Enhancing Technologies*. Springer, pp. 194–208.

Girling, C. Gray (1987). "Covert Channels in LAN's". In: *IEEE Transactions on Software Engineering* 13 (2), pp. 292–296.

Github (2019). *GitHub website*. URL: https://github.com.

Glöckner, Andreas, Susann Fiedler, and Frank Renkewitz (2018). "Belastbare und effiziente Wissenschaft. Strategische Ausrichtung von Forschungsprozessen als Weg aus der Replikationskrise". In: *Psychologische Rundschau* 69.1. in German, pp. 22–36.

Google Inc. (2015). *Google Scholar website*. URL: http://scholar.google.de.

Graf, T. (2003). *Messaging over IPv6 Destination Options*. URL: http://gray-world.net/papers/messip6.txt.

Gunadi, Hendra and Sebastian Zander (2017a). *Bro Covert Channel Detection (BroCCaDe) Framework: Design and Implementation*. Tech. rep. IT NSRG TR 20171117B. Murdoch University.

Gunadi, Hendra and Sebastian Zander (2017b). *Bro Covert Channel Detection (BroCCaDe) Framework: Scope and Background*. Tech. rep. IT NSRG TR 20171117A. Murdoch University.

Gunarathne, Thilina, Jarek Blaminsky, Edward Gordon, Puja Lalwani, Alfida Paiva, and Laxmi Subramanian (2015). *Hadoop MapReduce v2 Cookbook*. 2nd ed. Packt Publishing.

Haas, Herbert (2010). *Mausezahn*. URL: https://github.com/uweber/mausezahn.

Hammond, M. M., N. L. Neff, J. L. Farr, A. R. Schwall, and X. Y. Zhao (2011). "Predictors of Individual-Level Innovation at Work: A Meta-Analysis". In: *Psychology of Aesthetics Creativity and the Arts* 5.1, pp. 90–105.

Handel, T. G. and M. T. Sandford II (1996). "Hiding Data in the OSI Network Model". In: *Proc. First International Workshop on Information Hiding*. London, UK: Springer, pp. 23–38.

Handley, M., V. Paxson, and C. Kreibich (2001). "Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics". In: *Proc. 10th USENIX Security Symposium*. Vol. 10, pp. 115–131.

Henninger, S. and V. Corrêa (2007). *Software Pattern Communities: Current Practices and Challenges*. Tech. rep. TR-UNL-CSE-2007-0015. University of Nebraska.

Herzberg, A. and H. Shulman (2013). "Limiting MitM to MitE Covert Channels". In: *Proc. 2013 International Conferences on Availability, Reliability and Security (ARES'13)*. IEEE, pp. 236–241.

Houmansadr, A., C. Brubaker, and V. Shmatikov (2013). "The Parrot is Dead: Observing Unobservable Network Communications". In: *Proc. 34th IEEE Symposium on Security and Privacy*. IEEE, pp. 65–79.

Houmansadr, Amir, Negar Kiyavash, and Nikita Borisov (2009). "RAINBOW: A Robust And Invisible Non-Blind Watermark for Network Flows". In: *Proceedings of 16th Annual Network & Distributed System Security Symposium (NDSS)*. URL: http://www.hatswitch.org/~nikita/papers/rainbow-ndss09.pdf.

Howard, T. J., S. J. Culley, and E. Dekoninck (2008). "Describing the creative design process by the integration of engineering design and cognitive psychology literature". In: *Design Studies* 29.2, pp. 160–180.

Hu, W.-M. (1991). "Reducing Timing Channels with Fuzzy Time". In: *Proc. 1991 Symposium on Security and Privacy*. IEEE, pp. 8–20.

Janicki, Artur (2016). "Pitch-based Steganography for Speex Voice Codec". In: *Security and Communication Networks* 9.15, pp. 2923–2933. DOI: 10.1002/sec.1428.

Jankowski, B., W. Mazurczyk, and K. Szczypiorski (2010). "Information Hiding Using Improper frame padding". In: *Proc. 14th International Telecommunications Network Strategy and Planning Symposium (NETWORKS)*, pp. 1–6.

– (2013). "PadSteg: introducing inter-protocol steganography". In: *Telecommunication Systems* 52.2, pp. 1101–1111. ISSN: 1572-9451. DOI: 10.1007/s11235-011-9616-z. URL: https://doi.org/10.1007/s11235-011-9616-z.

Ji, L., H. Liang, Y. Song, and X. Niu (2009). "A Normal-Traffic Network Covert Channel". In: *Proc. International Conference on Computational Intelligence and Security*, pp. 499–503.

Ji, L., Y. Fan, and C. Ma (2010). "Covert channel for local area network". In: *Proc. Int. Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pp. 316–319.

Kang, M. H. and I.S. Moskowitz (1993). "A Pump for Rapid, Reliable, Secure Communication". In: *Proc. 1st ACM Conference on Computer and Communication Security*, pp. 119–129.

Katzenbeisser, Stefan and Fabien A. P. Petitcolas (2002). "Defining Security in Steganographic Systems". In: *Proc. Security and Watermarking of Multimedia Contents IVSPIE 2002*. Vol. 4675. DOI: http://dx.doi.org/10.1117/12.465313.

Kaufman, J. C. and R. A. Beghetto (2009). "Beyond big and little: The four c model of creativity". In: *Review of General Psychology* 13.1, pp. 1–12.

– (2013). "Do people recognize the four Cs? Examining layperson conceptions of creativity". In: *Psychology of Aesthetics, Creativity, and the Arts* 7.3, pp. 229–236.

Keidel, Ralf, Steffen Wendzel, Sebastian Zillien, Eric S. Conner, and Georg Haas (2018). "WoDiCoF – A Testbed for the Evaluation of (Parallel) Covert Channel Detection Algorithms". In: *Journal of Universal Computer Science (J.UCS)* 24.5, pp. 556–576.

Kemmerer, R. A. (1983). "Shared resource matrix methodology: an approach to identifying storage and timing channels". In: *ACM Transactions on Computer Systems* 1.3, pp. 256–277. ISSN: 0734-2071. DOI: http://doi.acm.org/10.1145/357369.357374.

Khor, K. A. and L.-G. Yu (2016). "Influence of international co-authorship on the research citation impact of young unviersities". In: *Scientometrics* 107, pp. 1095–1110.

Klauser, Tobias, Daniel Borkmann, et al. (2017). *Trafgen (part of netsniff-ng toolkit)*. URL: http://netsniff-ng.org.

Korpela, E., D. Werthimer, D. Anderson, J. Cobb, and M. Leboisky (2001). "SETI@home-massively distributed computing for SETI". In: *Computing in Science Engineering* 3.1, pp. 78–83. ISSN: 1521-9615. DOI: 10.1109/5992.895191.

Kosten, J. (2016). "A classification of the use of research indicators". In: *Scientometrics* 108, pp. 457–464.

Kraetzer, C., J. Dittmann, A. Lang, and T. Kuehne (2006). "WLAN Steganography: A First Practical Review". In: *Proc. 8th Workshop on Multimedia and Security (MMSEC'06)*, pp. 17–22.

Krause, D. E. (2013). *Kreativität, Innovation und Entrepreneurship*. (in German). Wiesbaden: Springer.

Kundur, Deepa and Kamran Ahsan (2003). "Practical Internet steganography: data hiding in IP". In:

Lampson, B. W. (1973). "A Note on the Confinement Problem". In: *Comm. ACM* 16.10, pp. 613–615.

Laursen, Keld and Ammon J. Salter (2014). "The paradox of openness: Appropriability, external search and collaboration". In: *Research Policy* 43.5. Open Innovation: New Insights and Evidence, pp. 867 –878. ISSN: 0048-7333. DOI: 10.1016/j.respol.2013.10.004.

Lewandowski, G., N. Lucena, and Steve C. (2007). "Analyzing Network-Aware Active Wardens in IPv6". In: *Proc. Information Hiding*. Vol. 4437. LNCS. Springer, pp. 58–77.

Li, X., Y. Zhang, F.T. Chong, and B.Y. Zhao (2011). *A Covert Channel Analysis of a Real Switch*. Tech. rep. Dep. of Computer Science, University of California, Santa Barbara.

Lin, Thomas (2012). *Cracking Open the Scientific Process*. URL: http://www.nytimes.com/2012/01/17/science/open-science-challenges-journal-tradition-with-web-collaboration.html.

Liu, Yali, Dipak Ghosal, Frederik Armknecht, Ahmad-Reza Sadeghi, Steffen Schulz, and Stefan Katzenbeisser (2010). "Robust and undetectable steganographic timing channels for iid traffic". In: *International Workshop on Information Hiding*. Springer, pp. 193–207.

Llamas, D., C. Allison, and A. Miller (2005). "Covert channels in Internet protocols: A survey". In: *Proc. 6th Annual Postgraduate Symp. Convergence of Telecommunications, Networking and Broadcasting (PGNET 2005)*.

Lubacz, J., W. Mazurczyk, and K. Szczypiorski (2014). "Principles and overview of network steganography". In: *IEEE Communications Magazine* 52.5, pp. 225–229.

Lubart, T. L. (2001). "Models of the creative process: Past, present and future". In: *Creativity Research Journal* 13.3-4, pp. 295–308.

Lucena, N., J. Pease, P. Yadollahpour, and S. J. Chapin (2004). "Syntax and Semantics-Preserving Application-Layer Protocol Steganography". In: *Proceedings of 6th Information Hiding Workshop*. Toronto, Canada.

Lucena, N., G. Lewandowski, and S. Chapin (2006). "Covert Channels in IPv6". In: *Proc. 5th International Workshop on Privacy Enhancing Technologies (PET 2005)*. Vol. 3856. LNCS. Springer, pp. 147–166.

Luo, X., E. W. W. Chan, and R. K. C. Chang (2007). "Cloak: A Ten-Fold Way for Reliable Covert Communications". In: *Computer Security – ESORICS 2007*. Ed. by Joachim Biskup and Javier López. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 283–298. ISBN: 978-3-540-74835-9.

Marcus, A. (2004). "Patterns within patterns". In: *interactions* 11.2, pp. 28–34.

May, Daniel and Paul Taylor (2003). "Knowledge Management with Patterns". In: *Communications of the ACM* 46.7, pp. 94–99.

Mazurczyk, W. and E. Rzeszutko (2015). "Security – a perpetual war: lessons from nature". In: *IEEE IT Professional*.

Mazurczyk, W. and K. Szczypiorski (2008). "Steganography of VoIP Streams". In: *Proc. OnTheMove Federated Conferences and Workshops (OTM) 2008, Part II – The 3rd International Symposium on Information Security (IS'08)*. Vol. 5332. LNCS. Springer-Verlag Berlin Heidelberg, pp. 1001–1018.

– (2012). "Evaluation of steganographic methods for oversized IP packets". In: *Telecommunication Systems* 49.2, pp. 207–217.

Mazurczyk, W., M. Smolarczyk, and K. Szczypiorski (2011). "Retransmission Steganography and Its Detection". In: *Soft Computing* 15.3, pp. 505–515.

Mazurczyk, Wojciech (2013). "VoIP Steganography and Its Detection – A Survey". In: *ACM Comput. Surv.* 46.2, 20:1–20:21. ISSN: 0360-0300. DOI: 10.1145/2543581.2543587. URL: http://doi.acm.org/10.1145/2543581.2543587.

Mazurczyk, Wojciech and Luca Caviglione (2015). "Information Hiding as a Challenge for Malware Detection". In: *Security Privacy, IEEE* 13.2, pp. 89–93. ISSN: 1540-7993. DOI: 10.1109/MSP.2015.33.

Mazurczyk, Wojciech and Józef Lubacz (2010). "LACK—a VoIP steganographic method". In: *Telecommunication Systems* 45.2, pp. 153–163. ISSN: 1572-9451. DOI: 10.1007/s11235-009-9245-y. URL: https://doi.org/10.1007/s11235-009-9245-y.

Mazurczyk, Wojciech and Steffen Wendzel (2018). "Information Hiding: Challenges for Forensic Experts". In: *Communications of the ACM (CACM)* 61.1, pp. 86–94. DOI: 10.1145/3158416. URL: https://dl.acm.org/citation.cfm?id=3158416.

Mazurczyk, Wojciech, Paweł Szaga, and Krzysztof Szczypiorski (2014). "Using Transcoding for Hidden Communication in IP Telephony". In: *Multimedia Tools Appl.* 70.3, pp. 2139–2165. ISSN: 1380-7501. DOI: 10.1007/s11042-012-1224-8. URL: http://dx.doi.org/10.1007/s11042-012-1224-8.

Mazurczyk, Wojciech, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, and Krzysztof Szczypiorski (2016a). *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. Wiley-IEEE.

Mazurczyk, Wojciech, Steffen Wendzel, Ignacio Azagra Villares, and Krzysztof Szczypiorski (2016b). "On importance of steganographic cost for network steganography". In: *Security and Communication Networks (SCN)* 9.8, pp. 781–790.

Mazurczyk, Wojciech, Steffen Wendzel, and Krzysztof Cabaj (2018). "Towards Deriving Insights into Data Hiding Methods Using Pattern-based Approach". In: *Proc. Second International Workshop on Criminal Use of Information Hiding (CUING 2018)*. ACM, 10:1–10:10.

Mazurczyk, Wojciech, Steffen Wendzel, Mehdi Chourib, and Jörg Keller (2019a). "Countering adaptive network covert communication with dynamic wardens". In: *Future Generation Computer Systems* 94, pp. 712–725. DOI: 10.1016/j.future.2018.12.047.

Mazurczyk, Wojciech, Przemysław Szary, Steffen Wendzel, and Luca Caviglione (2019b). "Towards Reversible Storage Network Covert Channels". In: 69:1–69:8.

McLennan, M. and R. Kennell (2010). "HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering". In: *Computing in Science Engineering* 12.2, pp. 48–53. ISSN: 1521-9615. DOI: 10.1109/MCSE.2010.41.

Meadows, C. and I. S. Moskowitz (1996). "Covert channels–A context-based view". In: *Proc. Information Hiding Workshop*. Vol. 1174. LNCS. Springer Berlin Heidelberg.

Mendeley Ltd. (2015). *Mendeley Website*. https://www.mendeley.com.

Meyer, B. (2011). "The Nastiness Problem in Computer Science". In: *Communications of the ACM Blog*. URL: http://cacm.acm.org/blogs/blog-cacm/123611-the-nastiness-problem-in-computer-science/fulltext.

Moghaddam, H., B. Li, M. Derakhshani, and I. Goldberg (2012). "SkypeMorph: Protocol Obfuscation for Tor Bridges". In: *CCS*.

Mogul, J. C. (2013). "Towards more constructive reviewing of CS papers". In: *ACM SIGCOMM Computer Communication Review* 43.3, pp. 90–94. ISSN: 0146-4833. DOI: 10.1145/2500098.2500112.

Mooney, R. L. (1963). "A conceptual model for integrating four approaches to the identification of creative talent". In: *Scientific Creativity: Its Recognition and Development*. Ed. by C. W. Taylor and F. Barron. New York: Wiley.

Moskowitz, I.S. and M.H. Kang (1994). "Covert channels-here to stay?" In: *Computer Assurance, 1994. COMPASS '94 Safety, Reliability, Fault Tolerance, Concurrency and Real Time, Security. Proceedings of the Ninth Annual Conference on*, pp. 235–243. DOI: 10.1109/CMPASS.1994.318449.

Muchene, D. N., K. Luli, and C. A. Shue (2013). "Reporting Insider Threats via Covert Channels". In: *2013 IEEE Security and Privacy Workshops*, pp. 68–71.

Mumford, M. D. and S. B. Gustafson (1988). "Creativity Syndrome: Integration, application, and innovation". In: *Psychological Bulletin* 103.1, pp. 27–43.

Mumford, M. D., W. A. Baughman, M. A. Maher, D. P. Costanza, and E. P. Supinski (1997). "Process-Based Measures of Creative Problem-Solving Skills: IV. Category Combination". In: *Creativity Research Journal* 10.1, pp. 59–71.

Mumford, M. D., S. Connelly, and B. Gaddis (2003). "How creative leaders think". In: *The Leadership Quarterly* 14.4-5, pp. 411–432.

Murdoch, S. J. (2007). "Covert channel vulnerabilities in anonymity systems". PhD thesis. University of Cambridge (Computer Laboratory).

Murdoch, S. J. and S. Lewis (2005). "Embedding Covert Channels into TCP/IP". In: *Proc. Information Hiding Conference 2005*. Vol. 3727. LNCS. Springer, pp. 247–261.

Murdoch, Steven J. and George Danezis (2005). "Low-cost traffic analysis of Tor". In: *Proceedings of IEEE Symposium on Security and Privacy*, pp. 183–195. ISBN: 0-7695-2339-0.

Murdoch, Steven J. and Piotr Zielinski (2004). "Covert Channels for Collusion in Online Computer Games". In: *Proc. 6th Information Hiding Workshop*. Toronto, Ontario, Canada: Springer.

Nattkemper, T. W. (2012). "Are we ready for Science 2.0?" In: *International Conference on Knowledge Management and Information Sharing*. Ed. by Kecheng Liu and Joaquim Filipe. Barcelona, Spain, pp. 302–306.

Object Management Group (OMG) (2010). *Unified Modeling Language (OMG UML), Infrastructure, Version 2.3*.

Ogurtsov, N., H. Orman, R. Schroeppel, et al. (1996). *Covert Channel Elimination Protocols*. Tech. rep. Department of Computer Science, University of Arizona.

Oldham, G. R. and A. Cummings (1996). "Employee creativity: Personal and contextual factors at work". In: *Academy of Management Journal* 39.3, pp. 607–634.

Overleaf (2019). *Overleaf*. URL: https://www.overleaf.com.

P. R. Gallagher, Jr., ed. (1993). *A Guide to Understanding Covert Channel Analysis of Trusted Systems (NCSC-TG-030)*. National Computer Security Center.

Palmer, C. (2015). "Berufsbezogene Kreativitätsdiagnostik. Entwicklung und Validierung eines Verfahrens zur Erfassung der personalen Voraussetzungen von Innovationen". (in German). PhD thesis. Stuttgart: Justus-Liebig-Universität Giessen.

Palmer, C., B. Cesinger, P. Gellèri, D. Putsch, and J. Winzen (2015). "Psychometrical testing of entrepreneurial creativity". In: *International Journal of Entrepreneurial Venturing* 7.2, pp. 194–210.

Parnas, D. L. (1972). "On the Criteria to Be Used in Decomposing Systems into Modules". In: *Commun. ACM* 15.12, pp. 1053–1058. ISSN: 0001-0782. DOI: 10.1145/361598.361623. URL: http://doi.acm.org/10.1145/361598.361623.

Patuck, R. and J. Hernandez-Castro (2013). "Steganography using the Extensible Messaging and Presence Protocol (XMPP)". In: *CoRR* abs/1310.0524.

Petitcolas, Fabien AP, Ross J Anderson, and Markus G Kuhn (1999). "Information hiding—a survey". In: *Proceedings of the IEEE* 87.7, pp. 1062–1078.

Polak, Lukasz and Zbigniew Kotulski (2010). "Sending hidden data through WWW pages: detection and prevention". In: *Engineering Transactions* 58.1-2, pp. 75–89.

Porras, P. A. and R. A. Kemmerer (1991). "Covert Flow Trees: A Technique for Identifying and Analyzing Covert Storage Channels". In: *Proc. IEEE Symposium on Security and Privacy*, pp. 36–51.

Priem, J. and B. M. Hemminger (2010). "Scientometrics 2.0: Toward new metrics of scholarly impact on the social Web". In: *First Monday* 15.7.

Proctor, N. E. and P. G. Neumann (1992). "Architectural Implications of Covert Channels". In: *Proc. of 15th National Computer Security Conference*, pp. 28–43.

Public Library of Science (2019). *PLOS Website*. URL: https://www.plos.org.

Ray, B. and S. Mishra (2008). "A Protocol for Building Secure and Reliable Covert Channel". In: *Proc. 6th Annual Conference on Privacy, Security and Trust (PST 2008)*. IEEE, pp. 246–253.

ResearchGate (2019). *ResearchGate website*. URL: http://www.researchgate.net.

Rhodes, M. (1961). "An analysis of creativity". In: *Phi Delta Kappa* 42, pp. 305–310.

Rios, Ruben, JoseA. Onieva, and Javier Lopez (2012). "HIDE_DHCP: Covert Communications through Network Configuration Messages". English. In: *Information Security and Privacy Research*. Ed. by Dimitris Gritzalis, Steven Furnell, and Marianthi Theoharidou. Vol. 376. IFIP Advances in Information and Communication Technology. Berlin Heidelberg: Springer, pp. 162–173.

Roure, David De, Carole Goble, and Robert Stevens (2009). "The design and realisation of the Virtual Research Environment for social sharing of workflows". In: *Future Generation Computer Systems* 25.5, pp. 561 –567. ISSN: 0167-739X. DOI: 10.1016/j.future.2008.06.010.

Rowland, C. H. (1997). "Covert Channels in the TCP/IP protocol suite". In: *First Monday* 2.5.

Runco, M. A. (2006). "Introduction to the Special Issue: Divergent Thinking". In: *Creativity Research Journal* 18.3, pp. 249–250.

Rutkowska, J. (2004). *The implementation of passive covert channels in the Linux kernel*. Speech held at the 21st Chaos Communication Congress, Berlin, Germany. URL: http://events.ccc.de/congress/2004/fahrplan/files/319-passive-covert-channels-slides.pdf.

Sadeghi, A.-R., S. Schulz, and V. Varadharajan (2012). "The Silence of the LANs: Efficient Leakage Resilience for IPsec VPNs". In: *Computer Security – ESORICS 2012*. Vol. 7459. LNCS. Springer Berlin Heidelberg, pp. 253–270.

Satyanarayanan, M. (2018). "Saving Software from Oblivion". In: *IEEE Spectrum* 2018.10, pp. 36–41.

Schmidt, Sabine S., Wojciech Mazurczyk, Jörg Keller, and Luca Caviglione (2017). "A New Data-Hiding Approach for IP Telephony Applications with Silence Suppression". In: *Proceedings of the 12th International Conference on Availability, Reliability and Security*. ARES '17. Reggio Calabria, Italy: ACM, 83:1–83:6. ISBN: 978-1-4503-5257-4. DOI: 10.1145/3098954.3106066. URL: http://doi.acm.org/10.1145/3098954.3106066.

Schneier, Bruce (2019). *Fraudulent Academic Papers*. URL: https://www.schneier.com/blog/archives/2019/05/fraudulent_acad.html.

Schuler, H. and Y. Görlich (2007). *Kreativität. Ursachen, Messung, Förderung und Umsetzung in Innovation*. (in German). Göttingen: Hogrefe.

Seffah, A. (2010). "The evolution of design patterns in HCI: from pattern languages to pattern-oriented design". In: *Proceedings of the 1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems*. PEICS '10. Berlin, Germany: ACM, pp. 4–9.

Servetto, S. D. and M. Vetterli (2001). "Communication using phantoms: covert channels in the Internet". In: *Proc. 2011 IEEE Int. Symp. on Information Theory*, p. 229.

Shachtman, N. (2010). "FBI: Spies Hid Secret Messages on Public Websites". In: *Wired*. URL: http://www.wired.com/2010/06/alleged-spies-hid-secret-messages-on-public-websites/.

Shah, G., A. Molina, and M. Blaze (2006). "Keyboards and Covert Channels". In: *Proc. 15th USENIX Security Symposium*. USENIX Association, pp. 59–75.

Shen, J., S. Qing, Q. Shen, and L. Li (2005). "Optimization of Covert Channel Identification". In: *Proc. 3rd IEEE Int. Security in Storage Workshop (SISW'05)*. IEEE, pp. 95–108.

Siaterlis, C., B. Genge, and M. Hohenadel (2013). "EPIC: A Testbed for Scientifically Rigorous Cyber-Physical Security Experimentation". In: *IEEE Transactions on Emerging Topics in Computing*, pp. 319–330.

Sieberg, D. (2001). "Bin Laden exploits technology to suit his needs". In: *CNN*. URL: http://edition.cnn.com/2001/US/09/20/inv.terrorist.search/.

Silvia, P. J., R. E. Beaty, E. C. Nusbaum, K. M. Eddington, H. Levin-Aspenson, and T. R. Kwapil (2014). "Everyday Creativity in Daily Life: An Experience-Sampling Study of "Little c" Creativity". In: *Psychology of Aesthetics, Creativity, and the Arts* 8.2, pp. 183–188.

Simmons, Gustavus J. (1984). "The Prisoners' Problem and the Subliminal Channel". In: *Advances in Cryptology: Proceedings of Crypto 83*. Boston, MA: Springer, US, pp. 51–67.

Simonton, D. K. (2004). *Creativity in Science. Chance, Logic, Genius, and Zeitgeist*. Cambridge University Press.

– (2011). "Creativity". In: *The Oxford Handbook of Positive Psychology*. Ed. by S. J. Lopez and C. R. Snyder. 2nd. Oxford University Press. Chap. 24, pp. 261–269.

– (2014). "More method in the mad-genius controversy: A historiometric study of 204 historic creators". In: *Psychology of Aesthetics, Creativity, and the Arts* 8.1, pp. 53–61.

Smith, A. J. (1990). "The task of the referee". In: *IEEE Computer* 23.4, pp. 65–71.

Smith, G. J. W. (2005). "How Should Creativity Be Defined?" In: *Creativity Research Journal* 17, pp. 293–295.

Snort Project (2012). *Snort Users Manual 2.9.3*.

Sohn, T., J. Seo, and J. Moon (2003). "A Study on the Covert Channel Detection of TCP/IP Header Using Support Vector Machine". In: *5th International Conference on Information and Communications Security*. Huhehaote, China, pp. 313–324. ISBN: 3-540-20150-5.

Stefano Avallone, Antonio Pescapè and Giorgio Ventre (2003). "Distributed Internet Traffic Generator (D-ITG): analysis and experimentation over heterogeneous networks". In: *Int. Conference on Network Protocols (ICNP'03) poster proceedings*.

Steinebach, M., E. Hauer, and P. Wolf (2007). "Efficient Watermarking Strategies". In: *Third International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution (AXMEDIS'07)*, pp. 65–71. DOI: 10.1109/AXMEDIS.2007.9.

Steinebach, Martin, Andre Ester, and Huajian Liu (2018). "Channel Steganalysis". In: *Proceedings of the 13th International Conference on Availability, Reliability and Security*. ARES 2018. Hamburg, Germany: ACM, 9:1–9:8. ISBN: 978-1-4503-6448-5. DOI: 10.1145/3230833.3233266. URL: http://doi.acm.org/10.1145/3230833.3233266.

Stødle, D. (2011). *Ping Tunnel*. URL: http://www.cs.uit.no/~daniels/PingTunnel/index.html.

Subhedar, Mansi S. and Vijay H. Mankar (2014). "Current status and key issues in image steganography: A survey". In: *Computer Science Review* 13-14, pp. 95 –113. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2014.09.001. URL: http://www.sciencedirect.com/science/article/pii/S1574013714000136.

Szczypiorski, Krzysztof (2012). "A performance analysis of HICCUPS—a steganographic system for WLAN". In: *Telecommunication Systems* 49.2, pp. 255–259. ISSN: 1572-9451. DOI: 10.1007/s11235-010-9363-6. URL: https://doi.org/10.1007/s11235-010-9363-6.

Tanenbaum, A. S. and H. Bos (2015). *Modern Operating Systems*. 4th ed. Pearson.

The PubPeer Foundation (2019). *PubPeer*. URL: https://www.pubpeer.com.

Tidwell, J. (2009). *Designing Interfaces: Patterns for Effective Interaction Design*. O'Reilly Media.

Tiedtke, T., T. Krach, and C. Märtin (2005). "Multi-Level Patterns for the Planes of User Experience". In: *Theories Models and Processes in HCI* 4.

Toxboe, Anders (2019). *User Interface Design patterns*. URL: http://ui-patterns.com/patterns.

Trabelsi, Z. and I. Jawhar (2010). "Covert File Transfer Protocol Based on The IP Record Route Option". In: *Journal of Information Assurance and Security (JIAS)* 5.1.

Tsiatsikas, Zisis, Marios Anagnostopoulos, Georgios Kambourakis, Sozon Lambrou, and Dimitris Geneiatakis (2015). "Hidden in Plain Sight. SDP-Based Covert Channel for Botnet Communication". In: *Trust, Privacy and Security in Digital Business*. Ed. by Simone Fischer-Hübner, Costas Lambrinoudakis, and Javier López. Vol. 9264. LNCS. Springer International

Publishing, pp. 48–59. ISBN: 978-3-319-22905-8. DOI: 10.1007/978-3-319-22906-5_4. URL: http://dx.doi.org/10.1007/978-3-319-22906-5_4.

Tumoian, E. and M. Anikeev (2005). "Network Based Detection of Passive Covert Channels in TCP/IP". In: *1st IEEE LCN Workshop on Network Security*, pp. 802–809.

Tuptuk, Nilufer and Stephen Hailes (2015). "Covert channel attacks in pervasive computing". In: *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 236–242. DOI: 10.1109/PERCOM.2015.7146534.

Van Duyne, D.K., J.A. Landay, and J.I. Hong (2007). *The Design of Sites: Patterns For Creating Winning Web Sites*. Prentice Hall.

Van Noorden, Richard (2014). "Online collaboration: Scientists and the social network". In: *Nature (News Feature)* 512, pp. 126–129.

Velino, Aleksandar, Aleksandra Mileva, Steffen Wendzel, and Wojciech Mazurczyk (2019). "Covert Channels in the MQTT-based Internet of Things". In: *IEEE Access*. in press.

WAND group (2000). *NZIX data set*. URL: https://wand.net.nz/wits/nzix/2/.

Ward, T. B. (2004). "Cognition, creativity, and entrepreneurship". In: *Journal of Business Venturing* 19.2, pp. 173–188.

Welie, M. van (2001). *Patterns in Interaction Design*. URL: http://www.welie.com.

Wendzel, Steffen (2019). "Protocol-independent Detection of 'Messaging Ordering' Network Covert Channels". In: *Proc. Third International Workshop on Criminal Use of Information Hiding (CUING 2019)*, 63:1–63:8.

Wendzel, Steffen and Jörg Keller (2012a). "Preventing Protocol Switching Covert Channels". In: *International Journal On Advances in Security* 5.3 and 4, pp. 81–93.

Wendzel, Steffen and Jörg Keller (2011). "Low-attention forwarding for mobile network covert channels". In: *Proc. Communications and Multimedia Security*. Vol. 7025. LNCS. Springer, pp. 122–133.

– (2012b). "Preventing Protocol Switching Covert Channels". In: *International Journal On Advances in Security* 5.3 and 4, pp. 81–93.

– (2012c). "Systematic Engineering of Control Protocols for Covert Channels". In: *Proc. 13th IFIP TC 6/TC 11 International Conference on Communications and Multimedia Security (CMS 2012)*. Vol. 7394. LNCS. Springer, pp. 131–144.

– (2014). "Hidden and Under Control: A Survey and Outlook on Covert Channel-internal Control Protocols". In: *Annals of Telecommunications, ANTE* 69.7, pp. 417–430.

Wendzel, Steffen and Wojciech Mazurczyk (2016). "Poster: An Educational Network Protocol for Covert Channel Analysis Using Patterns". In: *Proc. 23rd ACM Conference on Computer and Communications Security (CCS)*. ACM, pp. 1739–1741.

Wendzel, Steffen and Carolin Palmer (2015). "Creativity in Mind: Evaluating and Maintaining Advances in Network Steganographic Research". In: *Journal of Universal Computer Science (J.UCS)* 21.12, pp. 1684–1705. URL: http://www.jucs.org/jucs_21_12/creativity_in_mind_evaluating.

Wendzel, Steffen and Sebastian Zander (2012). "Detecting Protocol Switching Covert Channels". In: *37th IEEE Conference on Local Computer Networks (LCN)*. Clearwater, Florida, USA: IEEE, pp. 280–283.

Wendzel, Steffen, Benjamin Kahler, and Thomas Rist (2012). "Covert Channels and Their Prevention in Building Automation Protocols: A Prototype Exemplified Using BACnet". In: *Proc. 2012 Int. Conf. Green Computing and Communications (GreenCom)*. IEEE, pp. 731–736.

Wendzel, Steffen, Viviane Zwanger, Michael Meier, and Sebastian Szlósarczyk (2014a). "Envisioning Smart Building Botnets". In: *Proc. Sicherheit 2014*. Vol. 228. LNI, pp. 319–329.

Wendzel, Steffen, Wojciech Mazurczyk, Luca Caviglione, and Michael Meier (2014b). "Hidden and Uncontrolled – On the Emergence of Network Steganographic Threats". In: *Information Security Solutions Europe (ISSE'14)*. Securing Electronic Business Processes. Springer-Vieweg, pp. 123–133.

Wendzel, Steffen, Sebastian Zander, Bernhard Fechner, and Christian Herdin (2015). "Pattern-based Survey and Categorization of Network Covert Channel Techniques". In: *Computing Surveys (CSUR)* 47.3.

Wendzel, Steffen, Wojciech Mazurczyk, and Sebastian Zander (2016). "Unified Description for Network Information Hiding Methods". In: *Journal of Universal Computer Science* 22.11, pp. 1456–1486.

Wendzel, Steffen, Wojciech Mazurczyk, and Gerog Haas (2017a). "Don't You Touch My Nuts: Information Hiding in Cyber Physical Systems". In: *2017 IEEE Security and Privacy Workshops (SPW)*, pp. 29–34. DOI: 10.1109/SPW.2017.40.

Wendzel, Steffen, Luca Caviglione, Wojciech Mazurczyk, and Jean-Francois Lalande (2017b). "Network Information Hiding and Science 2.0: Can it be a Match?" In: *Int. Journal of Electronics and Telecommunications* 62.2, pp. 217–222.

Wendzel, Steffen, Wojciech Mazurczyk, and Georg Haas (2017c). "Steganography for Cyber-physical Systems". In: *Journal of Cyber Security and Mobility (JCSM)* 6.2, pp. 105–126.

Wendzel, Steffen, Daniela Eller, and Wojciech Mazurczyk (2018). "One Countermeasure, Multiple Patterns: Countermeasure Variation for Covert Channels". In: *Proc. Central European Security Conference (CECC'18)*. ACM, 1:1–1:6.

Wendzel, Steffen, Florian Link, Daniela Eller, and Wojciech Mazurczyk (2019). "Detection of Size Modulation Covert Channels Using Countermeasure Variation". In: *Journal of Universal Computer Science (J.UCS)*. in press.

West, Joel, Ammon Salter, Wim Vanhaverbeke, and Henry Chesbrough (2014). "Open innovation: The next decade". In: *Research Policy* 43.5. Open Innovation: New Insights and Evidence, pp. 805 –811. ISSN: 0048-7333. DOI: 10.1016/j.respol.2014.03.001.

White, Steve R. (1989). "Covert distributed processing with computer viruses". In: *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, pp. 616–619.

Wolf, M. (1989). "Covert channels in LAN protocols". In: *Proc. Local Area Network Security*. Vol. 396. LNCS. Springer, pp. 89–101.

Yarochkin, F. V., S.-Y. Dai, C.-H. Lin, et al. (2008). "Towards Adaptive Covert Communication System". In: *Proc. 14th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2008)*. IEEE Computer Society, pp. 153–159.

Yoder, J. and J. Barcalow (1997). "Architectural Patterns for Enabling Application Security". In: *Proceedings of the 4th Conference of Pattern Languages of Programs*.

Zander, S., G.J. Armitage, and P.A. Branch (2006). "Covert Channels in the IP time to live field". In: *Australian Telecommunication Networks and Applications Conference (ATNAC 2006)*, pp. 298–302.

Zander, S., G. J. Armitage, and P. Branch (2007a). "A survey of covert channels and countermeasures in computer network protocols". In: *IEEE Communications Surveys and Tutorials* 9 (3), pp. 44–57.

Zander, S., G. Armitage, and P. Branch (2007b). "Covert channels and countermeasures in computer network protocols". In: *IEEE Communications Magazine* 45.12, pp. 136–142.

Zander, Sebastian and Grenville Armitage (2008). *CCHEF–Covert channels evaluation framework design and implementation*. Tech. rep. Swinburne University of Technology. Centre for Advanced Internet Architectures.

Zander, Sebastian and Steven J. Murdoch (2008). "An Improved Clock-skew Measurement Technique for Revealing Hidden Services". In: *Proceedings of Usenix Security*. URL: http://www.usenix.org/events/sec08/tech/full_papers/zander/zander_html.

Zander, Sebastian, Grenville Armitage, and Philip A. Branch (2008). "Covert Channels in Multiplayer First Person Shooter Online Games". In: *Proceedings of 33rd Annual IEEE Conference on Local Computer Networks (LCN)*. Montreal, Canada. URL: http://caia.swin.edu.au/cv/szander/publications/lcn2008_preprint.pdf.

– (2011). "Stealthier Inter-packet Timing Covert Channels". In: *IFIP Networking*. Berlin Heidelberg: Springer, pp. 458–470.

Zhiyong, C. and Z. Yong (2009). "Entropy based taxonomy of network convert channels". In: *Proc. 2nd Int. Conf. on Power Electronics and Intelligent Transportation System (PEITS)*, pp. 451–455.

Zhou, J. and C. E. Shalley (2003). "Research on employee creativity: A critical review and directions for future research". In: *Research in Personnel and Human Resources Management* 22, pp. 165–217.

Zillien, Sebastian and Steffen Wendzel (2018). "Detection of covert channels in TCP retransmissions". In: *Proc. 23rd Nordic Conference on Secure IT Systems (NordSec)*. Vol. 11252. LNCS. Springer, pp. 203–218.

Zooniverse (2019). *Galaxy Zoo*. URL: http://www.galaxyzoo.org.

Zou, Xin-guang, Qiong Li, Sheng-He Sun, and Xiamu Niu (2005). "The Research on Information Hiding Based on Command Sequence of FTP Protocol". In: *Proc. 9th Int. Conf. on Knowledge-Based Intelligent Information and Engineering Systems (KES 2005), Part III*. Vol. 3683. LNCS. Springer Berlin Heidelberg, pp. 1079–1085.

Zseby, Tanja, Félix Iglesias Vázquez, Valentin Bernhardt, Davor Frkat, and Robert Annessi (2016). "A Network Steganography Lab on Detecting TCP/IP Covert Channels". In: *IEEE Transactions on Education* 59.3, pp. 224–232. ISSN: 0018-9359. DOI: 10.1109/TE.2016.2520400.

Abstract

Network information hiding is the research discipline that deals with the concealment of network transmissions or their characteristics. It serves as an umbrella for multiple research domains, namely network covert channel research, network steganography research, and traffic obfuscation research. The focus of this thesis lies primarily on network steganography and network covert channel research.

This thesis was motivated by the fact that network information hiding requires a better scientific foundation. When the author started to work on this thesis, scientific re-inventions of hiding techniques were common (similar or equal techniques were published under different names by different scientific sub-communities). This is, at least partially, rooted in the non-unified terminology of the domain, and in the sheer fact that the ever-increasing number of publications in the domain is hardly knowable. Moreover, experimental results and descriptions for hiding techniques are hardly comparable as there is no unified standard for describing them. This is a contrast to other scientific domains, such as Chemistry, were (de facto) standards for experimental descriptions are common. Another problem is that experimental results are not replicated while other scientific domains have shown that replication studies are a necessity to ensure the quality of scientific results. Finally, there is an imbalance between known hiding techniques and their countermeasures: not enough countermeasures are known to combat all known hiding techniques.

To address these issues, this thesis motivates and proposes methodological adjustments in network information hiding and lays the foundation for an improved fundamental terminology and taxonomy.

Moreover, hiding techniques are surveyed and summarized in the form of abstract descriptions, called hiding patterns, which form an extensible taxonomy. These hiding patterns are then used as a tool to evaluate the novelty of research contributions in a scientific peer-review process. Afterwards, this thesis addresses the problem of inconsistent descriptions of hiding techniques by proposing a unified description method for the same, including hiding patterns as a core component of every description. This thesis also introduces the *WoDiCoF* testbed as a framework to perform replication studies.

Afterwards, the concept of countermeasure variation is introduced to address the problem of not having countermeasures available for certain hiding patterns. Finally, the proposed pattern-based taxonomy is enhanced to demonstrate the extensibility of the taxonomy and to integrate payload-based hiding techniques which were not foreseen in the earlier version of the taxonomy.

Steffen Wendzel received his Ph.D. (Dr. rer. nat.) from the University of Hagen in 2013. Afterwards he led a security research team at Fraunhofer FKIE, Bonn. In 2016, he became a professor for computer networks and information security at the University of Applied Sciences, Worms. Steffen Wendzel is (co-)author of over 150 publications, including six books. His website: *www.wendzel.de*.