Managing Your Kleptographic Subscription Plan

George Teşeleanu[1,2]

[1] Department of Computer Science
"Al.I.Cuza" University of Iaşi 700506 Iaşi, Romania,
george.teseleanu@info.uaic.ro
[2] Advanced Technologies Institute
10 Dinu Vintilă, Bucharest, Romania
tgeorge@dcti.ro

Abstract. In the classical kleptographic business models, the manufacturer of a device D is paid either in advance or in installments by a malicious entity to backdoor D. Unfortunately, these models have an inherent high risk for the manufacturer. This translates in high costs for clients. To address this issue, we introduce a subscription based business model and tackle some of the technical difficulties that arise.

1 Introduction

Kleptographic attacks have been introduced by Young and Yung [22–26] and are a combination of subliminal channels with public key cryptography. The scope of these attacks is to leak either confidential messages or private keys though a system's outputs without the owner's knowledge. In recent years, this research area has been revitalized and backdooring methodologies can be found for symmetric key primitives [7, 8, 10], hash functions [5, 14], pseudo-random number generators [11, 12] or digital signatures [6, 21]. Also, a series of countermeasures have been developed [6, 15, 18, 19].

One of the classical business models for kleptographic attacks is the following: a client[3] C pays up front a manufacturer M, whom will later implement a certain backdoor in a tamper proof device[4] and deliver that device to a victim. This model puts the manufacturer at an advantage, because he can charge the customer and not implement the requested backdoor. Since this transaction is illegal, the customer can not file a complain and legally retrieve his money. Thus, this might scare off some of the potential clients.

Another classical model is the following: a client pays the manufacturer half the money up front and the rest after checking the correctness of the backdoor. If the manufacturer does not take certain precautions, then the client is at an advantage. For example, C checks the correctness of the backdoor, but fails to pay the second installment. This can be easily avoided if a backdoor deactivation method is put in place by M[5]. A possible deactivation strategy is for M to send D a special input that instructs the device to erase all incriminating evidence. A similar approach is used in [10, 14] to trigger backdoors.

Both classical approaches have an inherent risk for the manufacturer: the client can easily prove that M backdoored D either by decrypting all the messages send through that device or by revealing the private keys stored in D. Thus, to make the risk worth while the manufacturer must charge C a high embedding fee. This will certainly scare away certain resource constrained clients (*e.g.* small businesses that do not have the resources of a large corporation). To address this issue, we introduce a subscription based model suitable for the ElGamal encryption algorithm.

Our model draws inspiration from the subscription services offered by companies like Netflix [2], Amazon [3] and HBO [4]. These companies give access to streaming content in exchange for a monthly pay. In our case, a client pays for a backdoor that gives him access to a limited number of private messages. Subsequently, the client has to renew his subscription. This balances the risk and reward factors for the manufacturer[6]

[3] by definition a malicious entity

[4] In [8] is noted that complex open-source software (*e.g.* OpenSSL) is also vulnerable to these attacks.

[5] As in the previous model, the transaction is illegal and thus, M can not take legal action against C.

[6] M is exposed only for a limited period of time

and, in consequence, M can lower embedding fees. A risk still remains: no guarantees of output delivery for the clients. But, this is minimum in a subscription based model because the goal of the manufacturer is to keep clients satisfied, so they further renew their subscription[7].

Compared to the classical models, our proposed model has a different issue that needs to be tackled. Clients want access to their services as soon as they pay. But, illegal transactions mostly use cryptocurrencies [9] and the average confirmation time for this type of transactions is large in some cases (*e.g.* for Bitcoin, it takes on average an hour per transaction [1]). Thus, to give the manufacturer sufficient time for deactivating the backdoor[8] if the transaction is not valid, we employ a mechanism similar to time-lock puzzles [17] .

Note that generic kleptographic countermeasures [15, 18, 19] can protect tamper proof device's users against our proposed mechanisms. Unfortunately, unless users do not explicitly require the implementation of these defences, a manufacturer is not obliged to deploy them. Thus, M is free to implement any kleptographic mechanism.

Structure of the paper. Notations and definitions are presented in Section 2. The core of the paper consists of Section 3 and contains a series of kleptographic subscriptions that fit different scenarios. We conclude in Section 4.

2 Preliminaries

Notations. Throughout the paper, the subset $\{1, \ldots, n\} \in \mathbb{N}$ is denoted by $[1, n]$. The action of selecting a random element x from a sample space X is denoted by $x \xleftarrow{\$} X$, while $x \leftarrow y$ represents the assignment of value y to variable x. The probability of the event E to happen is denoted by $Pr[E]$. To ease description, we use the notation C^n_k to denote binomial coefficients.

2.1 Security Assumptions

Definition 1 (Pseudorandom Function - PRF). *A function $F : \mathbb{G} \times [1, n] \to S$ is a PRF if:*

- *Given a key $K \in \mathbb{G}$ and an input $X \in [1, n]$ there is an efficient algorithm to compute $F_K(X) = F(X, K)$.*
- *Let A be a PPT algorithm with access to an oracle $\mathcal{O}$ that returns 1 if $\mathcal{O} = F_K(\cdot)$. The PRF-advantage of A, defined as*

$$ADV^{\text{PRF}}_F(A) = \left| Pr[A^{F_K(\cdot)} = 1 | K \xleftarrow{\$} \mathbb{G}] - Pr[A^{F(\cdot)} = 1 | F \xleftarrow{\$} \mathcal{F}] \right|$$

must be negligible for any PPT algorithm A, where $\mathcal{F} = \{F : [1, n] \to S\}$.

Definition 2 (Pseudorandom Permutation - PRP). *A PRF $P : \mathbb{G} \times [1, n] \to [1, n]$ is a PRP if P is one-to-one and $\mathcal{F}$ from Definition 1 is changed into $\mathcal{F} = \{F : [1, n] \to [1, n] \mid F$ is one-to-one$\}$. The PRP-advantage of A is denoted $ADV^{\text{PRP}}_P(A)$.*

Definition 3 (Decisional Diffie-Hellman - DDH). *Let $\mathbb{G}$ be a cyclic group of order q, g a generator of $\mathbb{G}$. Let A be a PPT algorithm which returns 1 on input (g^x, g^y, g^z) if $g^{xy} = g^z$. We define the advantage*

$$ADV^{\text{DDH}}_{\mathbb{G},g}(A) = |Pr[A(g^x, g^y, g^z) = 1 | x, y \xleftarrow{\$} \mathbb{Z}^*_q, z \leftarrow xy] - Pr[A(g^x, g^y, g^z) = 1 | x, y, z \xleftarrow{\$} \mathbb{Z}^*_q]|.$$

If $ADV^{\text{DDH}}_{\mathbb{G},g}(A)$ is negligible for any PPT algorithm A, we say that the Decisional Diffie-Hellman problem is hard in $\mathbb{G}$.

[7] Cheating a client will only bring M a small amount of revenue.

[8] by means of special triggers

2.2 Public Key Encryption

Definition 4 (Public Key Encryption - PKE**).** *A Public Key Encryption (PKE) scheme consists of four PPT algorithms: ParamGen, KeyGen, Encrypt and Decrypt. The first one takes as input a security parameter and outputs the system parameters. Using these parameters, the second algorithm generates the public key and the matching secret key. The public key together with the Encrypt algorithm are used to encrypt a message m. Using the secret key, the last algorithm decrypts any ciphertext encrypted using the matching public key.*

Remark 1. For simplicity, public parameters will further be implicit when describing an algorithm.

ElGamal Encryption. The ElGamal encryption scheme was first described in [13] and later generalized in [16]. It can be proven that the generalized ElGamal encryption scheme is secure in the standard model under the DDH assumption [20]. We further describe the generalized version of the scheme and refer to it simply as the ElGamal encryption scheme (EG).

ParamGen(λ)*:* Generate a large prime number q, such that $q \geq 2^\lambda$. Choose a cyclic group $\mathbb{G}$ of order q and let g be a generator of the group. Output the public parameters $pp = (q, g, \mathbb{G})$.

KeyGen(pp)*:* Choose $x \xleftarrow{\$} \mathbb{Z}_q^*$ and compute $y \leftarrow g^x$. Output the public key $pk = y$. The secret key is $sk = x$.

Encryption(m, pk)*:* To encrypt a message $m \in \mathbb{G}$, first generate a random number $k \xleftarrow{\$} \mathbb{Z}_q^*$. Then compute the values $c \leftarrow g^k$ and $d \leftarrow m \cdot y^k$. Output the pair (c, d).

Decryption(c, d, sk)*:* To recover the original message compute $m \leftarrow d \cdot c^{-x}$.

2.3 SETUP Attacks

Definition 5 (Secretly Embedded Trapdoor with Universal Protection - SETUP**).** *A Secretly Embedded Trapdoor with Universal Protection (SETUP) is an algorithm that can be inserted in a system such that it leaks encrypted confidential messages to an attacker through the system's outputs. Encryption of the messages is performed using an asymmetric encryption scheme. It is assumed that the corresponding decryption function is accessible only to the attacker.*

Definition 6 (SETUP **indistinguishability -** IND-SETUP**).** *Let C_0 be a black-box system that uses a secret key sk. Let $\mathcal{AE}$ be the PKE scheme used by a SETUP mechanism as defined above, in Definition 5. We consider C_1 an altered version of C_0 that contains a SETUP mechanism based on $\mathcal{AE}$. Let A be a PPT algorithm which returns 1 if it detects that C_0 is altered. We define the advantage*

$$ADV_{C_0,C_1}^{\textit{IND-SETUP}}(A) = |Pr[A^{C_1(\cdot)}(\lambda) = 1] - Pr[A^{C_0(\cdot)}(\lambda) = 1]|.$$

If $ADV_{\mathcal{AE},C_0,C_1}^{\textit{IND-SETUP}}(A)$ is negligible for any PPT algorithm A, we say that C_0 and C_1 are polynomially indistinguishable.

All kleptographic subscriptions presented from now on are implemented in a device D. The owner of the device is denoted by V and we assume that he is in possession of his secret key. Note that V thinks that D contains an implementation of the ElGamal scheme as described in Section 2.2. When one of the original ElGamal algorithms is not modified by the SETUP attack, the scheme will be omitted when presenting the respective attack.

Throughout the paper, when presenting kleptographic subscriptions, we make use of the following additional algorithms:

- *Device's/Manufacturer's/Customer's KeyGen* – used by the device/manufacturer/customer to generate its/his keys;
- *Token* – used by the customer/manufacturer to extract the access token;
- *Extract* – used by the customer to recover the messages sent by V.

The previously mentioned algorithms are not implemented in D. For simplicity, kleptographic parameters will further be implicit when describing a scheme.

3 Kleptographic Subscriptions

3.1 Free Subscription

The first type of subscription (denoted by FS) is an analog of public television channels. Thus, anyone who is in possession of the transmitted ciphertexts can decrypt them after a certain amount of traffic has been sent. This protocol will form the basis for the mechanisms presented in Sections 3.2 and 3.3.

Although, this kind of subscription does not bring any revenue, it can still be useful in certain situations. For example, a disgruntled employee can embed it in the source code of certain products before leaving the company. Then, he can anonymously point out that the respective company implemented backdoors in their products. The scope of this scenario is to damage the company's reputation.

Let n be the maximum number of messages that a client needs to wait before recovering all of V's communications. Also, let $F : \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_q^*$. When searching for the access token, we make use of an auxiliary function *Check* that returns true if the decrypted message is correct. We further present the algorithms for the free subscription SETUP attack.

Device's KeyGen(pp): Choose $x_D \xleftarrow{\$} \mathbb{Z}_q^*$ and $p \xleftarrow{\$} [0,n]$. Output the device's secret key $sk_D = (x_D, p)$.

Encryption Sessions: The possible encryption sessions performed by D are described below. Let $i \neq p$.

Encryption$_i$ (m_i, pk, sk_D)*:* To encrypt a message $m_i \in \mathbb{G}$, first compute $k_i \leftarrow F(g^{x_D}, i)$. Then compute the values $c_i \leftarrow g^{k_i}$ and $d_i \leftarrow m_i \cdot y^{k_i}$. Output the pair (c_i, d_i).

Encryption$_p$ (m_p, pk, sk_D)*:* To encrypt a message $m_p \in \mathbb{G}$, compute the values $c_p \leftarrow g^{x_D}$ and $d_p \leftarrow m_p \cdot y^{x_D}$. Output the pair (c_p, d_p). Erase p from D's memory.

Token$(c_1, d_1, \ldots, c_n, d_n, pk)$*:* Let $i = 1$. Compute $k_{i+1} \leftarrow F(c_i, i \bmod n + 1)$, $m_{i+1} \leftarrow d_{i+1} \cdot y^{-k_{i+1}}$ and $i \leftarrow i + 1$, until *Check*$(m_i) = \texttt{true}$. Output the token p.

The ith Extract(c_i, d_i, p)*:* To recover the ith message compute $k_i \leftarrow F(c_p, i)$ and $m_i \leftarrow d_i \cdot y^{-k_i}$.

Remark 2. It is easy to see that message m_p can only be retrieved by the recipient.

We further state the security margin without proof due to its similarity to the more involved proof of Theorem 2.

Theorem 1. *If F is a* PRF *and $i \in [1, p-1]$ then EG and FS are* IND-SETUP*. Formally, let A be an efficient PPT* IND-SETUP *adversary. There exists an efficient algorithm B such that*

$$ADV^{\text{IND-SETUP}}_{EG,\ FS}(A) \leq 2ADV^{\text{PRF}}_{F}(B).$$

3.2 Paid Subscription

In this subsection, we describe a kleptographic analogue of payed television (denoted by PS). Thus, C pays M for a session's access token, that only M can extract from D. Note that these tokens are unique per session. So, a group of users can pay for only one token and all of them will have access to that session's private messages. Although this can be considered cheating, it is also a reality in other systems (*e.g.* paying for a Netflix account and sharing the credentials with one's friends). We will rectify this problem in the next subsection.

Let t be a security parameter and $P : \mathbb{G} \times [1, n] \rightarrow [1, n]$. After the first message is transmitted the manufacturer will send the clients a set of t positions p_j needed to compute the access token. Note that M has a window of at least $t-1$ messages to receive his payments. If one payment is declined, M can deactivate the backdoor before the t-th message has been issued. A downside of this scheme is that if one of the clients fails to pay for the token, then he deprives all users of their access.

We further state one session of the protocol. After a predetermined number of messages (greater than n) have elapsed, D can generate new keys and start a new session.

Manufacturer's KeyGen(pp): Choose $x_M \xleftarrow{\$} \mathbb{Z}_q^*$ and compute $y_M \leftarrow g^{x_M}$. Output the manufacturer's public key $pk_M = y_M$. The secret key is $sk_M = x_M$. Store pk_M in D's internal memory.

Device's KeyGen(pp): Choose $k_0 \xleftarrow{\$} \mathbb{Z}_q^*$. For each $j \in [1, t]$ compute $p_j \leftarrow P(y_M^{k_0}, j)$ and choose $x_j \xleftarrow{\$} \mathbb{Z}_q^*$. Compute $x_D \leftarrow x_1 + \ldots + x_t$. Store the device's secret key $sk_D = (k_0, p_1, \ldots, p_t, x_1, \ldots, x_t, x_D)$.

Encryption Sessions: The possible encryption sessions performed by D are described below. Let $i \in [0, n]$ and $i \neq p_j$, for each $j \in [1, t]$. The algorithm for $Encryption_i$ are identical to the public subscription and thus are omitted.

$Encryption_0(m_0, pk)$: To encrypt a message $m_0 \in \mathbb{G}$ compute the values $c_0 \leftarrow g^{k_0}$ and $d_0 \leftarrow m_0 \cdot y^{k_0}$. Output the pair (c_0, d_0). Erase k_0 from D's memory.

$Encryption_{p_j}(m_{p_j}, pk, sk_D)$: To encrypt a message $m_{p_j} \in \mathbb{G}$, compute the values $c_{p_j} \leftarrow g^{x_j}$ and $d_{p_j} \leftarrow m_{p_j} \cdot y^{x_j}$. Output the pair (c_{p_j}, d_{p_j}). Erase (p_j, x_j) from D's memory.

$Token(c_0, sk_M)$: For each $j \in [1, t]$ compute $p_j \leftarrow P(c_0^{x_M}, j)$. Output the token $p = (p_1, \cdots, p_t)$.

The ith $Extract(c_i, d_i, p)$: To recover the ith message compute $c_p \leftarrow c_{p_1} \cdot \ldots \cdot c_{p_t}$ and $k_i \leftarrow F(c_p, i)$ and $m_i \leftarrow d_i \cdot y^{-k_i}$.

Remark 3. It is easy to see that messages $m_0, m_{p_1}, \ldots, m_{p_t}$ can not be retrieved by the customers.

Theorem 2. *If* DDH *is hard in $\mathbb{G}$, P is a* PRP*, F is a* PRF *and $(C_n^t)^{-1}$ is negligible then EG and PS are* IND-SETUP*. Formally, let A be an efficient PPT* IND-SETUP *adversary. There exist three efficient algorithms B_1, B_2 and B_3 such that*

$$ADV_{EG,\ PS}^{\text{IND-SETUP}}(A) \leq 2ADV_{\mathbb{G},g}^{DDH}(B_1) + 2ADV_{P}^{PRP}(B_2) + 2ADV_{F}^{PRF}(B_3) + (C_t^n)^{-1}.$$

Proof. Let A be an IND-SETUP adversary trying to distinguish between EG and PS. We show that A's advantage is negligible. We construct the proof as a sequence of games in which all the required changes are applied to PS. Let W_i be the event that A wins game i.

Game 0. The first game is identical to the IND-SETUP game[9]. Thus, we have

$$|2Pr[W_0] - 1| = ADV_{\text{EG,PS}}^{\text{IND-SETUP}}(A). \qquad (1)$$

[9] as in Definition 6

Game 1. In this game, instead of using $y_M^{k_0}$ as a key to P we use $r_P \xleftarrow{\$} \mathbb{G}$. More precisely, for each $j \in [1, t]$ we compute $p_j \leftarrow P(r_P, j)$. Since this is the only change between *Game 0* and *Game 1*, A will not notice the difference assuming the DDH assumption holds. Formally, this means that there exists an algorithm B_1 such that

$$|Pr[W_0] - Pr[W_1]| = ADV_{\mathbb{G},g}^{\text{DDH}}(B_1). \tag{2}$$

Game 2. Since P is a PRP then we can choose $p_j \xleftarrow{\$} [1, n]$, without A detecting the change. Formally, this means that there exists an algorithm B_2 such that

$$|Pr[W_1] - Pr[W_2]| = ADV_P^{\text{PRP}}(B_2). \tag{3}$$

Game 3. In each $Encryption_{p_j}$ algorithm we make the change $c_{p_j} \leftarrow g^{k_j}$ and $d_{p_j} \leftarrow m_{p_j} y^{k_j}$, where $k_j \xleftarrow{\$} \mathbb{Z}_q^*$. Since k_js and x_js have the same distribution, and the b_js are uniformly distributed in $[1, n]$, then A can only detect the change using a brute-force attack[10]. Formally, we have

$$|Pr[W_2] - Pr[W_3]| = (C_t^n)^{-1}. \tag{4}$$

Game 4. The last change we make is $k_i \xleftarrow{\$} \mathbb{Z}_q^*$. Adversary A will not notice the difference, since F is a PRF. Formally, this means that there exists an algorithm B_3 such that

$$|Pr[W_3] - Pr[W_4]| = ADV_P^{\text{PRF}}(B_3). \tag{5}$$

The changes made to PS in *Game 1* $-$ *Game 4* transformed it into EG. Thus, we have

$$Pr[W_4] = 1/2. \tag{6}$$

Finally, the statement is proven by combining the equalities (1) $-$ (6). □

3.3 Targeted Subscription

As mentioned in the previous subsection, a coalition of clients can pay for only one token[11]. To avoid this problem we bind a specific session to a certain client. We could not find a method that allows multiple bindings per session. We further present the proposed solution for binding users and sessions (denoted by TS).

Customer's KeyGen(pp): Choose $x_C \xleftarrow{\$} \mathbb{Z}_q^*$ and compute $y_C \leftarrow g^{x_C}$. Output the customer's public key $pk_C = y_C$. The secret key is $sk_C = x_C$. Store pk_C in D's internal memory.

Encryption Sessions: The possible encryption sessions performed by D are described below. Let $i \in [0, n]$ and $i \neq p_j$, for each $j \in [1, t]$. The algorithms for $Encryption_0$ and $Encryption_{p_j}$ are identical to the paid subscription and thus are omitted.

$Encryption_i(m_i, pk, pk_C, sk_D)$: To encrypt a message $m_i \in \mathbb{G}$, first compute $k_i \leftarrow F(y_C^{x_D}, i)$. Then compute the values $c_i \leftarrow g^{k_i}$ and $d_i \leftarrow m_i \cdot y^{k_i}$. Output the pair (c_i, d_i).

[10] *i.e.* by trying each t-combination c_{try} of c_is, until on input c_{try} the *Extract* algorithm outputs a message m such that $Check(m) =$ `true`.

[11] further used by the whole group to access messages

The ith Extract(c_i, d_i, p)*:* To recover the ith message compute $c_p \leftarrow c_{p_1} \cdot \ldots \cdot c_{p_t}$ and $k_i \leftarrow F(c_p^{x_C}, i)$ and $m_i \leftarrow d_i \cdot y^{-k_i}$.

Theorem 2 assures us that the client has negligible probability of reading V's messages without M's help. We further prove a similar result for any PPT IND-SETUP adversaries.

Theorem 3. *If* DDH *is hard in* $\mathbb{G}$*,* P *is a* PRP *and* F *is a* PRF *then* EG *and* TS *are* IND-SETUP*. Formally, let* A *be an efficient PPT* IND-SETUP *adversary. There exist three efficient algorithms* B_1*,* B_2 *and* B_3 *such that*

$$ADV^{\text{IND-SETUP}}_{EG,\ TS}(A) \leq 4ADV^{\text{DDH}}_{\mathbb{G},g}(B_1) + 2ADV^{\text{PRP}}_{P}(B_2) + 2ADV^{\text{PRF}}_{F}(B_3).$$

Proof. *Game 0 – Game 2* and *Game 4* are identical to the games presented in the proof of Theorem 2 and thus, are omitted. Since only the customer is in position of x_C, we can not use the strategy presented in Theorem 2, *Game 3*. Thus, we present a modified version of *Game 3*.

Game 3'. In this game, we replace $y_C^{x_D}$ by $r_F \xleftarrow{\$} \mathbb{Z}_q^*$. Due to the fact that DDH is hard in $\mathbb{G}$, A will not notice the change. Formally, this means that there exists an algorithm B_1' such that

$$|Pr[W_2] - Pr[W_{3'}]| = ADV^{\text{DDH}}_{\mathbb{G},g}(B_1'). \tag{7}$$

Finally, the statement is proven by combining the equalities (1) – (3) and (5) – (7). □

4 Conclusions

In this paper we introduced the concept of subscription based kleptographic services and tackled the technical challenges associated with this model. The pay-as-you-go approach leads to better costs for the clients and minimizes exposure risks for the manufacturer.

Open Problems. A couple of interesting open problems are the extension of subscription based services to digital signatures and the implementation of multi-targeted subscriptions for one session.

References

1. Bitcoin: Average Confirmation Time. https://www.blockchain.com/charts/avg-confirmation-time
2. Frequently Asked Questions About Netflix Billing. https://help.netflix.com/en/node/41049?ui_action=kb-article-popular-categories
3. How to Manage Your Prime Video Channel Subscriptions. https://www.amazon.com/gp/help/customer/display.html?nodeId=201975160
4. How to Order HBO: Subscriptios & Pricing Options. https://www.hbo.com/ways-to-get
5. Albertini, A., Aumasson, J.P., Eichlseder, M., Mendel, F., Schläffer, M.: Malicious Hashing: Eve's Variant of SHA-1. In: SAC 2014. Lecture Notes in Computer Science, vol. 8781, pp. 1–19. Springer (2014)
6. Ateniese, G., Magri, B., Venturi, D.: Subversion-Resilient Signature Schemes. In: ACM-CCS 2015. pp. 364–375. ACM (2015)
7. Bellare, M., Jaeger, J., Kane, D.: Mass-Surveillance without the State: Strongly Undetectable Algorithm-Substitution Attacks. In: ACM-CCS 2015. pp. 1431–1440. ACM (2015)
8. Bellare, M., Paterson, K.G., Rogaway, P.: Security of Symmetric Encryption Against Mass Surveillance. In: CRYPTO 2014. Lecture Notes in Computer Science, vol. 8616, pp. 1–19. Springer (2014)
9. Christin, N.: Traveling the Silk Road: A Measurement Analysis of a Large Anonymous Online Marketplace. In: WWW 2013. pp. 213–224. ACM (2013)
10. Degabriele, J.P., Farshim, P., Poettering, B.: A More Cautious Approach to Security Against Mass Surveillance. In: FSE 2015. Lecture Notes in Computer Science, vol. 9054, pp. 579–598. Springer (2015)
11. Degabriele, J.P., Paterson, K.G., Schuldt, J.C., Woodage, J.: Backdoors in Pseudorandom Number Generators: Possibility and Impossibility Results. In: CRYPTO 2016. Lecture Notes in Computer Science, vol. 9814, pp. 403–432. Springer (2016)

12. Dodis, Y., Ganesh, C., Golovnev, A., Juels, A., Ristenpart, T.: A Formal Treatment of Backdoored Pseudorandom Generators. In: EUROCRYPT 2015. Lecture Notes in Computer Science, vol. 9056, pp. 101–126. Springer (2015)
13. ElGamal, T.: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory **31**(4), 469–472 (1985)
14. Fischlin, M., Janson, C., Mazaheri, S.: Backdoored Hash Functions: Immunizing HMAC and HKDF. IACR Cryptology ePrint Archive **2018/362** (2018)
15. Hanzlik, L., Kluczniak, K., Kutyłowski, M.: Controlled Randomness - A Defense against Backdoors in Cryptographic Devices. In: MyCrypt 2016. Lecture Notes in Computer Science, vol. 10311, pp. 215–232. Springer (2016)
16. Menezes, A.J., Van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC press (1996)
17. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock Puzzles and Timed-release Crypto. Tech. rep. (1996)
18. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Cliptography: Clipping the power of kleptographic attacks. In: ASIACRYPT 2016. Lecture Notes in Computer Science, vol. 10032, pp. 34–64. Springer (2016)
19. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Generic Semantic Security against a Kleptographic Adversary. In: ACM-CCS 2017. pp. 907–922. ACM (2017)
20. Shoup, V.: Sequences of Games: A Tool for Taming Complexity in Security Proofs. IACR Cryptology ePrint Archive **2004/332** (2004)
21. Teşeleanu, G.: Unifying Kleptographic Attacks. In: NordSec 2018. Lecture Notes in Computer Science, vol. 11252, pp. 73–87. Springer (2018)
22. Young, A., Yung, M.: The Dark Side of “Black-Box” Cryptography or: Should We Trust Capstone? In: CRYPTO 1996. Lecture Notes in Computer Science, vol. 1109, pp. 89–103. Springer (1996)
23. Young, A., Yung, M.: Kleptography: Using Cryptography Against Cryptography. In: EUROCRYPT 1997. Lecture Notes in Computer Science, vol. 1233, pp. 62–74. Springer (1997)
24. Young, A., Yung, M.: The Prevalence of Kleptographic Attacks on Discrete-Log Based Cryptosystems. In: CRYPTO 1997. Lecture Notes in Computer Science, vol. 1294, pp. 264–276. Springer (1997)
25. Young, A., Yung, M.: Malicious Cryptography: Exposing Cryptovirology. John Wiley & Sons (2004)
26. Young, A., Yung, M.: Malicious Cryptography: Kleptographic Aspects. In: CT-RSA 2005, Lecture Notes in Computer Science, vol. 3376, pp. 7–18. Springer (2005)