Subverting Deniability

Marcel Armour and Elizabeth A. Quaglia[*]

Royal Holloway, University of London
`{marcel.armour.2017,Elizabeth.Quaglia}@rhul.ac.uk`

Abstract. Deniable public-key encryption (DPKE) is a cryptographic primitive that allows the sender of an encrypted message to later claim that they sent a different message. DPKE's threat model assumes powerful adversaries who can coerce users to reveal plaintexts; it is thus reasonable to consider other advanced capabilities, such as being able to subvert algorithms in a so-called Algorithm Substitution Attack (ASA). ASAs have been considered against a number of primitives including digital signatures, symmetric encryption and pseudo-random generators. However, public-key encryption has presented a less fruitful target, as the sender's only secrets are plaintexts and ASA techniques generally do not provide sufficient bandwidth to leak these.
In this article, we give a formal model of ASAs against DPKE, and argue that subversion attacks against DPKE schemes present an attractive opportunity for an adversary. Our results strengthen the security model for DPKE and highlight the necessity of considering subversion in the design of practical schemes.

Keywords: Cryptography · Deniable Encryption · Algorithm Substitution Attacks.

1 Introduction

Deniable public-key encryption (DPKE) is a primitive that allows a sender to successfully lie about which plaintext message was originally encrypted. In particular, suppose that Alice encrypts a plaintext m under some public key, using randomness r, to give ciphertext c which she sends to Bob. At some point in the future – perhaps Bob falls under suspicion – Alice is coerced to reveal the message she encrypted, together with the randomness she used. DPKE allows Alice to claim that she sent m^*, by providing r^* such that $\mathsf{enc}(m^*, r^*) = \mathsf{enc}(m, r)$. Beyond its immediate use case, deniable encryption finds applications in electronic voting, where deniability allows voters to cast their ballots without coercion and prevents vote-buying, as well as in secure multiparty computation.

The adversarial model for deniable encryption assumes strong adversaries that can coerce individuals to reveal messages they encrypted; it is thus reasonable to consider other advanced capabilities, such as the ability to subvert

algorithms. Powerful adversaries can insert unreliability into cryptography via external ('real-world') infrastructure: whether by influencing standards bodies to adopt 'backdoored' parameters, inserting exploitable errors into software implementations, or compromising supply chains to interfere with hardware. The Snowden revelations showed that this is indeed the case; see the survey by Schneier et al. [12] which provides a broad overview of cryptographic subversion, with case studies of known subversion attempts.

Prior work considering subversion has usually had the aim of exfiltrating secret keys (in the context of symmetric encryption and digital signatures). Berndt and Liśkiewicz [4] show that a generic ASA against an encryption scheme can only embed a limited number of bits per ciphertext. More concretely, they show that no universal and consistent[1] ASA is able to embed more than $\log(\kappa)$ bits of information into a single ciphertext in the random oracle model [4, Theorem 1.4], where κ is the key length of the encryption scheme. In the setting of symmetric key encryption, this is sufficient to successfully leak the secret key over multiple ciphertexts [3,2]. However, for asymmetric primitives, subverting ciphertexts to leak the encryption key makes little sense as it is public; leaking plaintext messages is not possible due to the limited bandwidth. Thus for generic ASAs against PKE, the best possible adversarial goal is to exfiltrate sufficient information to compromise confidentiality – knowledge of one bit of the underlying plaintext is sufficient for an adversary to break confidentiality in the sense of IND-CPA or IND\$.[2] But as Bellare et al. [3] argue, this is not an attractive goal for a mass surveillance adversary, who would rather recover plaintext messages.

Contributions In this article we sketch out an argument to show that ASAs against DPKE schemes present an attractive opportunity for an adversary. We refer the reader to the extended versionfor full details and an expanded argument. In this article, we begin by recalling notions of ASAs, including adversarial goals (undetectability and information exfiltration), and give an example of a generic ASA technique (rejection sampling). We then recall DPKE notions, including the formal definition, before applying ASA definitions to DPKE. This allows us to present our main contribution, namely a description of how an ASA against DPKE could be successfully realised to undermine deniability. In brief: an adversary who can subvert a DPKE scheme can transmit a commitment to the underlying plaintext using a subliminal channel. Later, the adversary can check the commitment against the message that the sender claims was sent.

---

[1] Here universal means that the ASA applies generically to any encryption scheme, and consistent essentially means that the ASA outputs genuine ciphertexts.

[2] Chen et al. [8] overcome these limitations by using non-generic techniques against KEM-DEM constructions to leak underlying plaintexts representing (session) keys.

Our work is the first to consider subverting deniable encryption,[3] and we establish formal models of the adversarial goals as well as security notions for such an attack. In the extended version, we also consider how to mitigate ASAs against deniable encryption.

2 Notions of Subversion Attacks

We consider subversions of cryptographic schemes implementing encrypted communication between two parties. Abstractly, we consider a scheme $\Pi = (\Pi.\mathsf{gen}, \{\Pi.\mathsf{S}^{(i)}\}_{0 \le i < n}, \Pi.\mathsf{R})$ consisting of three components: a key generation algorithm, together with a collection of $n \in \mathbb{N}_{>0}$ sender algorithms and a receiver algorithm $\Pi.\mathsf{R}$ representing decryption. We let $\Pi.\mathsf{S}^{(0)}$ represent encryption and write $\Pi.\mathsf{S} := \Pi.\mathsf{S}^{(0)}$. Our abstract treatment allows us to capture both PKE schemes and symmetric encryption (setting $k_\mathsf{S} = k_\mathsf{R}$). We give a generic syntax to the scheme Π as follows: Key generation $\Pi.\mathsf{gen}$ outputs a key pair $(k_\mathsf{S}, k_\mathsf{R}) \in \mathcal{K}_\mathsf{S} \times \mathcal{K}_\mathsf{R}$. Each sender algorithm $\Pi.\mathsf{S}^{(i)}$, for $0 \le i < n$, has associated randomness space $\mathcal{R}^{(i)}$ together with input and output spaces $\mathcal{X}^{(i)}, \mathcal{Y}^{(i)}$ (respectively) and takes as input a sender key $k_\mathsf{S} \in \mathcal{K}_\mathsf{S}$ $x \in \mathcal{X}^{(i)}$, outputting $y \in \mathcal{Y}^{(i)}$; we write $\mathcal{X} := \mathcal{X}^{(0)}, \mathcal{Y} := \mathcal{Y}^{(0)}$. We note that $\mathcal{X} \subsetneq \mathcal{X}'$; in particular, $\perp \in \mathcal{X}' \setminus \mathcal{X}$. The receiver algorithm takes as input a receiver key $k_\mathsf{R} \in \mathcal{K}_\mathsf{R}$ and $y \in \mathcal{Y}$, outputting $x \in \mathcal{X}'$; the special symbol $\perp$ is used to indicate failure. Lastly, we foreground the randomness used during encryption in our notation by writing $y \leftarrow \Pi.\mathsf{S}(k_\mathsf{S}, x; r)$ for some randomness space $\mathcal{R}$ where we split the input space accordingly $\mathcal{X} \cong \tilde{\mathcal{X}} \times \mathcal{R}$; dropping the last input is equivalent to $r \leftarrow_\$ \mathcal{R}$. This allows us to discuss particular values of r that arise during encryption.

Undetectable Subversion In a nutshell, a subversion is undetectable if distinguishers with black-box access to either the original scheme or to its subverted variant cannot tell the two apart. A subversion should exhibit a dedicated functionality for the subverting party, but simultaneously be undetectable for all others. This apparent contradiction is resolved by parameterising the subverted algorithm with a secret subversion key, knowledge of which enables the extra functionality. We denote the subversion key space with $\mathcal{I}_\mathsf{S}$.

Formally: a subversion of the sender algorithm $\Pi.\mathsf{S}$ of a cryptographic scheme consists of a finite index space $\mathcal{I}_\mathsf{S}$ and a family $\mathcal{S} = \{\mathsf{S}_i\}_{i \in \mathcal{I}_\mathsf{S}}$ of algorithms such that for all $i \in \mathcal{I}_\mathsf{S}$ the algorithm $\Pi.\mathsf{S}_i$ can syntactically replace the algorithm $\Pi.\mathsf{S}$. As a security property we also require that the observable behaviour of $\Pi.\mathsf{S}$ and $\Pi.\mathsf{S}_i$ be effectively identical (for uniformly chosen $i \in \mathcal{I}_\mathsf{S}$). This is formalised via the games $\mathrm{UDS}^0, \mathrm{UDS}^1$ in fig. 1 (left). For any adversary $\mathcal{A}$ we define the advantage $\mathsf{Adv}_\Pi^{\mathrm{uds}}(\mathcal{A}) := \left| \Pr[\mathrm{UDS}^1(\mathcal{A})] - \Pr[\mathrm{UDS}^0(\mathcal{A})] \right|$ and say that family $\mathcal{S}$

---

[3] Gunn et al. [10] consider circumventing cryptographic deniability, which is similar in spirit. However, their scenario is quite different: firstly, they consider deniable communication protocols (such as Signal). Secondly, they do not consider subversion attacks – instead, their scenario is logically equivalent to compromising the receiver.

undetectably subverts algorithm $\Pi.\mathsf{S}$ if $\mathsf{Adv}^{\mathrm{uds}}_{\mathsf{PKE}}(\mathcal{A})$ is negligibly small for all realistic $\mathcal{A}$.

Subliminal Information Exfiltration Abstractly, the aim of an adversary is to exfiltrate some subliminal information. In the context of prior work considering symmetric encryption, this information typically represents the secret key. We formalise this goal as the MR game in fig. 1 (centre), which assumes a passive attack in which the adversary eavesdrops on communication, observing the transmitted ciphertexts. We allow the adversary some influence over sender inputs, with the aim of closely modelling real-world settings. This influence on the sender inputs x is restricted by assuming a stateful 'message sampler' algorithm MS that produces the inputs to $\Pi.\mathsf{S}$ used throughout the game. For any message sampler MS and adversary $\mathcal{A}$ we define the advantage $\mathsf{Adv}^{\mathrm{mr}}_{\Pi,\mathrm{MS}}(\mathcal{A}) := \Pr[\mathsf{MR}(\mathcal{A})]$. We say that subversion family $\mathcal{S}$ is key recovering for passive attackers if for all practical MS there exists a realistic adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathrm{mr}}_{\Pi,\mathrm{MS}}(\mathcal{A})$ reaches a considerable value (e.g., 0.1).[4]

Game $\mathrm{UDS}^b(\mathcal{A})$	**Game** $\mathsf{MR}(\mathcal{A})$	**Proc** $\Pi.\mathsf{S}_i(k_{\mathsf{S}}, x, \mu)$
00 $i \leftarrow_{\$} \mathcal{I}_{\mathsf{S}}$	00 $i \leftarrow_{\$} \mathcal{I}_{\mathsf{S}}$	00 while $[t \neq \mu]$:
01 $\mathsf{S}^0 := \Pi.\mathsf{S}_i$	01 $(k_{\mathsf{S}}, k_{\mathsf{R}}) \leftarrow_{\$} \Pi.\mathsf{gen};\ \sigma \leftarrow \diamond$	01 $\quad r \leftarrow_{\$} \mathcal{R}$
02 $\mathsf{S}^1 := \Pi.\mathsf{S}$	02 $\mu' \leftarrow \mathcal{A}^{\mathrm{Send}}(i)$	02 $\quad y \leftarrow \Pi.\mathsf{S}_i(k_{\mathsf{S}}, x; r)$
03 $b' \leftarrow \mathcal{A}^{\mathrm{Send}}$	03 stop with $[\mu' = \mu]$	03 $\quad t \leftarrow F_i(y)$
04 stop with b'		04 return y
	Oracle $\mathrm{Send}(\alpha)$	
Oracle $\mathrm{Send}(k_{\mathsf{S}}, x)$	04 $(\sigma, x, \beta) \leftarrow \mathsf{MS}(\sigma, \alpha)$	**Proc** $\mathcal{A}(i)$
05 $y \leftarrow \mathsf{S}^b(k_{\mathsf{S}}, x)$	05 $y \leftarrow \Pi.\mathsf{S}_i(k_{\mathsf{S}}, x)$	05 pick any α
06 return y	06 return (y, β)	06 $(y, \beta) \leftarrow \mathrm{Send}(\alpha)$
		07 $\mu \leftarrow F_i(y)$
		08 return μ

Fig. 1. Left: Game UDS modelling sender subversion undetectability for a scheme Π. **Centre:** Game MR modelling key recoverability for passive adversaries. **Right:** rejection sampling subversion $\Pi.\mathsf{S}_i$ of encryption algorithm $\Pi.\mathsf{S}$ and corresponding message recovering adversary $\mathcal{A}$.

Generic Method: Rejection Sampling As an example, we describe a generic method to embed a subliminal message μ with $|\mu| = \ell_\mu$ into ciphertexts of an encryption scheme $\Pi.\mathsf{S}$. Essentially, when computing a ciphertext, the subverted algorithm uses rejection sampling to choose randomness that results in a ciphertext encoding the subliminal message. The subverted encryption algorithm $\Pi.\mathsf{S}_i$

---

[4] Our informal notions ('realistic' and 'practical') are easily reformulated in terms of probabilistic polynomial-time (PPT) algorithms. However, given that asymptotic notions don't reflect practice particularly well, we prefer to use the informal terms.

of a scheme Π is given in fig. 1 (right) together with the corresponding message recovery adversary $\mathcal{A}$. Subversion $\Pi.\mathsf{S}_i$ is parameterised by a large index space $\mathcal{I}$, a constant ℓ_μ and a PRF F_i. For the PRF we require that it be a family of functions $F_i \colon \mathcal{Y} \to \{0,1\}^{\ell_\mu}$ (that is: a pseudo-random mapping from the ciphertext space to the set strings of length ℓ_μ).

We note that the subverted encryption algorithm $\Pi.\mathsf{S}_i$ will resample randomness 2^{ℓ_μ} times on average. This means that longer messages result in exponentially slower running times of the algorithm; in practice, this means that the attack is limited to short messages (a few bits at most).

3 Deniable Public Key Encryption

DPKE allows a sender to lie about the messages that were encrypted. In particular, suppose that a user encrypts message m to obtain c which is sent to the recipient. DPKE allows the sender to choose a different message m^* and reveal fake randomness r^* which explains c as the encryption of m^*. Notice that this necessarily implies that the scheme cannot be perfectly correct as $\mathsf{dec}(\mathsf{enc}(m^*, r^*)) = m$. This counter-intuitive observation is resolved by noticing that for a given message m, there are 'sparse trigger' values r_i such that encrypting m with an r_i results in an incorrect ciphertext. Deniable public-key encryption schemes rely on the fact that finding such r_i should be easy with some trapdoor knowledge, and hard otherwise.

In this article we focus on non-interactive sender deniable public-key encryption, as introduced by Canetti et al. [5], who showed that a sender-deniable scheme can be used to construct receiver deniable (and thus bi-deniable) schemes. To date, no practical deniable scheme has been proposed. Either deniability is not practically achievable (a typical example is Canetti et al.'s scheme [5] whose ciphertexts grow inversely proportional to the deniability probability), or else the construction requires strong assumptions such as iO or functional encryption [11,6,9]. Recent work by Agrawal et al. [1] is promising in this regard, as their construction provides compact ciphertexts and is based on the security of Learning with Errors. Nevertheless, they require a running time that is inversely proportional to detection probability.

DPKE Definition A DPKE scheme $\mathsf{DE} = (\mathsf{DE.gen}, \mathsf{DE.enc}, \mathsf{DE.dec}, \mathsf{DE.Fake})$ consists of a tuple of algorithms together with key spaces $\mathcal{K}_\mathsf{S}, \mathcal{K}_\mathsf{R}$, randomness space $\mathcal{R}$, a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$.

- The key-generation algorithm $\mathsf{DE.gen}$ returns a pair $(pk, sk) \in \mathcal{K}_\mathsf{S} \times \mathcal{K}_\mathsf{R}$ consisting of a public key and a private key.
- The encryption algorithm $\mathsf{DE.enc}$ takes a public key pk, randomness $r \in \mathcal{R}$ and a message $m \in \mathcal{M}$ to produce a ciphertext $c \in \mathcal{C}$.
- The decryption algorithm $\mathsf{DE.dec}$ takes a private key sk and a ciphertext $c \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the rejection symbol $\perp \notin \mathcal{M}$.

– The faking algorithm DE.Fake takes public key pk, a pair of message and randomness m, r and fake message m^*, and outputs faking randomness $r^* \in \mathcal{R}$.

A scheme DE is correct and secure if the key generation, encryption and decryption algorithms considered as a PKE scheme (DE.gen, DE.enc, DE.dec) satisfy the standard notions of correctness and IND-CPA security properties of public-key encryption. We formalise the deniability of the scheme via the game INDEXP, the details of which are given in the extended paper and described briefly here. Essentially, the INDEXP game is an indistinguishability game in which a distinguisher must choose between two cases: INDEXP^0 represents the adversary's view of an honest encryption of m^*; INDEXP^1 represents the adversary's view when the sender lies about the underlying plaintext. The corresponding advantage is, for any distinguisher $\mathcal{A}$, given by

$$\mathsf{Adv}^{\mathsf{indexp}}_{\mathsf{DE}}(\mathcal{A}) := \left| \Pr[\mathsf{INDEXP}^0(\mathcal{A})] - \Pr[\mathsf{INDEXP}^1(\mathcal{A})] \right|.$$

Formal Definition of Subverted DPKE We note that a DPKE scheme satisfies the generic syntax introduced above in section 2, with key generation algorithm $\Pi.\mathsf{gen} = \mathsf{DE.gen}$, sender algorithms $(\Pi.\mathsf{S}_0, \Pi.\mathsf{S}_1) = (\mathsf{DE.enc}, \mathsf{DE.Fake})$ and receiver algorithm $\Pi.\mathsf{R} = \mathsf{DE.dec}$. We may thus apply the generic notions of subversion and undetectability introduced in section 2. In the extended version, we furthermore consider the game subINDEXP modelling the adversary's ability to compromise the deniability property of a subverted scheme.

4 Subverting DPKE

Now that we have introduced the notions of ASAs and DPKE, we are ready to discuss ASAs against DPKE. As we set out in the introduction, the idea is for the subverted DPKE scheme to commit to the actual message encrypted; this undermines the ability of the sender to later claim that they sent a different message. The most obvious approach is to subvert the scheme so that the randomness commits to the message. This way, when Alice is coerced by the adversary to reveal her message and randomness, the adversary is able to test whether this is the case. This is a feasible attack route and applies generically to any deniable encryption scheme. When Alice claims that she sent m^*, by providing r^* such that $\mathsf{enc}(m^*, r^*) = \mathsf{enc}(m, r)$, she would need r^* to commit to the message. This should be hard, as long as the commitment is provided by a cryptographically secure digital signature or even a MAC (with the authentication key hidden from Alice). This generic ASA applies to all DPKE schemes, as the security definition for DPKE requires Alice to produce explanatory randomness when coerced.

A second technique is to use subversion techniques (such as the rejection sampling approach given as an example in section 2) to embed a subliminal channel in ciphertexts, such that the channel transmits a commitment to the message. The generic rejection sampling technique is unable to provide enough bandwidth to transmit sufficiently long signatures to prevent Alice forging the

commitment, however non-generic techniques may be possible depending on the particular scheme and instantiation. Furthermore, we note that it is a feature of most proposed deniable encryption schemes that a large amount of randomness is consumed in the course of encryption, and that this randomness is sampled in chunks. This means that if the algorithms are considered in a non-black box fashion, then rejection sampling could potentially be used against each chunk of randomness resulting in a sufficiently large subliminal channel.

Lastly, there is another subversion approach that at first glance seems appealing, but which turns out to be unworkable; namely, to target the faking algorithm. A subverted faking algorithm $\mathsf{DE.Fake}_i(pk, m, r, m^*)$ could output subverted r^* which alerts the adversary to the fact that m^*, r^* are fake; for example, if r^* commits to the real message m. However, this fake randomness r^* still needs to be convincing from the point of view of the deniability of the scheme – the scheme's security properties should be maintained by the subversion, otherwise a detector playing the UDS game will be able to tell that the algorithm is subverted. In particular, r^* should satisfy $\mathsf{DE.enc}(pk, m^*, r^*) = c$. However, for a DPKE scheme there is no reason why this should hold for an arbitrary value of r^*. This approach does not seem to be workable without adding considerable structure to the subverted scheme that means it would be easily detected.[5]

5   Conclusions

Deniable communication is a subtle concept and it is unclear what it should mean 'in the real world'. Intuitively, the notion is clear: deniability should allow Alice to plausibly evade incrimination when communicating. However, the adversarial model and evaluation of real world protocols claiming deniability is not agreed upon (should Alice be able to claim that she did not participant in a particular communication?). Celi and Symeonidis [7] give an overview of the current state of play and a discussion of open problems. Deniable encryption is one particular primitive whose definition is widely agreed upon in the literature and for which the applications are clear (including in e-voting, multi-party computation and to protect against coercion). The threat model for deniable encryption usually considers an adversary who is willing to coerce users; in this work we extend the model to consider adversaries who also undermine deniability by using subversion attacks. This seems a reasonable additional assumption to make of an adversary who is willing to engage in coercion. We hope that our work helps to elucidate some of the issues involved in designing deniable schemes and refine the threat model for deniable encryption.

---

[5] As an interesting aside, the approach for iO deniability schemes is to hide an encoding of the faked ciphertext within randomness; the encryption algorithm first checks whether the randomness encodes a ciphertext c and if so outputs c; if not, it proceeds to encrypt the message. The security follows from the fact that iO obfuscates the inner working of the algorithm so that it appears as a black box. This results in large, structured randomness inputs which would seem to facilitate subversion.

References

1. Agrawal, S., Goldwasser, S., Mossel, S.: Deniable fully homomorphic encryption from learning with errors. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, vol. 12826, pp. 641–670. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84245-1_22

2. Armour, M., Poettering, B.: Subverting decryption in AEAD. In: Albrecht, M. (ed.) 17th IMA International Conference on Cryptography and Coding. Lecture Notes in Computer Science, vol. 11929, pp. 22–41. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-35199-1_2

3. Bellare, M., Jaeger, J., Kane, D.: Mass-surveillance without the state: Strongly undetectable algorithm-substitution attacks. In: Ray, I., Li, N., Kruegel, C. (eds.) ACM CCS 2015: 22nd Conference on Computer and Communications Security. pp. 1431–1440. ACM Press (Oct 2015). https://doi.org/10.1145/2810103.2813681

4. Berndt, S., Liskiewicz, M.: Algorithm substitution attacks from a steganographic perspective. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017: 24th Conference on Computer and Communications Security. pp. 1649–1660. ACM Press (Oct / Nov 2017). https://doi.org/10.1145/3133956.3133981

5. Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable encryption. In: Kaliski Jr., B.S. (ed.) Advances in Cryptology – CRYPTO'97. Lecture Notes in Computer Science, vol. 1294, pp. 90–104. Springer, Heidelberg (Aug 1997). https://doi.org/10.1007/BFb0052229

6. Canetti, R., Park, S., Poburinnaya, O.: Fully deniable interactive encryption. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part I. Lecture Notes in Computer Science, vol. 12170, pp. 807–835. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56784-2_27

7. Celi, S., Symeonidis, I.: The current state of denial. In: HotPETS (2020)

8. Chen, R., Huang, X., Yung, M.: Subvert KEM to break DEM: Practical algorithm-substitution attacks on public-key encryption. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 98–128. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64834-3_4

9. De Caro, A., Iovino, V., O'Neill, A.: Deniable functional encryption. In: Cheng, C.M., Chung, K.M., Persiano, G., Yang, B.Y. (eds.) PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 9614, pp. 196–222. Springer, Heidelberg (Mar 2016). https://doi.org/10.1007/978-3-662-49384-7_8

10. Gunn, L.J., Parra, R.V., Asokan, N.: Circumventing cryptographic deniability with remote attestation. Proceedings on Privacy Enhancing Technologies **2019**(3), 350–369 (Jul 2019). https://doi.org/10.2478/popets-2019-0051

11. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th Annual ACM Symposium on Theory of Computing. pp. 475–484. ACM Press (May / Jun 2014). https://doi.org/10.1145/2591796.2591825

12. Schneier, B., Fredrikson, M., Kohno, T., Ristenpart, T.: Surreptitiously weakening cryptographic systems. Cryptology ePrint Archive, Report 2015/097 (2015), https://eprint.iacr.org/2015/097