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 the ability to subvert algorithms in a so-called Algorithm Substitution Attack (ASA). An ASA replaces a trusted algorithm with a subverted version that undermines security from the point of view of the adversary while remaining undetected by users. 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 work, we show that subversion attacks against deniable encryption schemes present an attractive opportunity for an adversary. We note that whilst the notion is widely accepted, there are as yet no practical deniable PKE schemes; we demonstrate the feasibility of ASAs targeting deniable encryption using a representative scheme as a proof of concept. We also provide a formal model and discuss how to mitigate ASAs targeting deniable PKE schemes. Our results strengthen the security model for deniable encryption 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. [36] which provides a broad overview of cryptographic subversion, with some useful case studies detailing known subversion attempts.

The idea that an adversary may embed a backdoor or otherwise tamper with the implementation or specification of a cryptographic scheme or primitive predates the Snowden revelations, and was initiated in a line of work by Young and Yung that they named *kleptography* [41,42]. This area of study can be

* This is the extended version of an article accepted for presentation at ProvSec 2022 and publication in Provable and Practical Security, Volume 13600 of the Lecture Notes in Computer Science series.

** The research of Armour was supported by the EPSRC and the UK government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/P009301/1).

traced back to Simmons' work on *subliminal channels*, e.g. [37], undertaken in the context of nuclear non-proliferation during the Cold War. In the original conception [41], kleptography considered a saboteur who designs a cryptographic algorithm whose outputs are computationally indistinguishable from the outputs of an unmodified trusted algorithm. The saboteur's algorithm should leak private key data through the output of the system, which was achieved using the same principles as Simmons' earlier subliminal channels. Post-Snowden, work in this area was reignited by Bellare, Paterson and Rogaway (BPR) [7], who formalised study of so-called *Algorithm Substitution Attacks* (ASAs) through the specific example of symmetric encryption schemes. In abstract terms, the adversary's goal in an ASA is to create a subverted implementation of a scheme that breaks some aspect of security (such as IND-CPA in the case of encryption) while remaining undetected by the user.

Prior work considering subversion has usually aimed to exfiltrate secret keys (in the context of symmetric encryption and digital signatures). Berndt and Liśkiewicz [9] 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 [9, 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 ([7,6,2]). 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 or two bits of the underlying plaintext message is sufficient to allow an adversary to break confidentiality in the sense of IND-CPA or IND\$.[2] But as Bellare et al. [6] argue, this is not an attractive goal for a mass surveillance adversary, who would rather break confidentiality completely and recover plaintext messages.

1.1 Contributions

In this work we argue that subversion attacks against deniable PKE schemes present an attractive opportunity for an adversary. A subversion adversary is willing to undermine the security of cryptographic primitives by leveraging their influence in the real world; coercing individuals to reveal the plaintexts they sent (or treating the inability to do so as an admission of guilt) falls within this remit. We demonstrate that deniable PKE schemes are vulnerable to ASAs that allow an adversary to subvert the deniability guarantees. Our key insight is that by transmitting a commitment to the underlying plaintext using a subliminal channel, the adversary can compare the commitment to the message that the coerced user claims to have sent. We consider two avenues to transmit the subliminal channel: using standard ASA techniques from the literature to embed the channel in ciphertexts; and via the randomness that the user reveals when coerced. 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. Lastly, we consider how to mitigate subversion attacks targeting deniable encryption.

[1] Here universal means that the ASA applies generically to any encryption scheme, and consistent essentially means that the ASA outputs genuine ciphertexts. We note that the rejection sampling ASA (Section 3.3) is universal and consistent, whereas IV replacement attacks (e.g. as discussed in Section 4) are not, failing to be universal.

[2] Chen et al. [16] overcome these limitations by using non-generic techniques against KEM-DEM constructions to leak underlying plaintext messages representing (session) keys. Armour and Poettering [3] consider a different approach by subverting the decryption algorithm of a PKE scheme to leak the private key.

[3] Gunn et al. [26] consider circumventing cryptographic deniability, which is similar in spirit to our work. However, the scenario they consider is quite different: firstly, they consider deniable communication protocols (such as Signal). Secondly, they do not consider subverting algorithms – instead, they consider subverting the receiver's device to generate a non-repudiable transcript that incriminates Alice, using remote attestation. Logically, this is equivalent to (verifiably) compromising Bob.

1.2 Structure of the paper

In Section 2 we give standard definitions of symmetric encryption (Section 2.2), public-key encryption (Section 2.3) and digital signatures (Section 2.4). Section 3 introduces a generic syntax for subversion attacks against encryption schemes and provides notions of undetectability (Section 3.1) as well as adversarial goals (Section 3.2). We describe a generic ASA in Section 3.3, and give an overview of approaches to mitigate ASAs in Section 3.4.

Having introduced the concept of an ASA, we go on to show that subverting deniability is a well-defined concept. We first consider, as an illustrative case study, subverting symmetric deniability in Section 4. Section 5 discusses deniable public-key encryption schemes, giving standard definitions and notions of security (Section 5.1) as well as a brief survey of the literature and a description of the 'Parity Scheme' of Canetti et al. [12](Section 5.2). Section 6 introduces notions of subverted deniability, including adversarial goals (Section 6). As a proof of concept, we show that the parity scheme is easily subverted (Section 6.1). We indicate approaches to mitigate subversion of deniable schemes in Section 6.2 and give our conclusions in Section 7.

2 Preliminaries and Standard Definitions

2.1 Notation

We refer to an element $x \in \{0,1\}^*$ as a string, and denote its length by $|x|$. The set of strings of length l is denoted $\{0,1\}^l$. By ε we denote the empty string. For $x \in \{0,1\}^*$ we let $x[i]$ denote the i-th bit of x, with the convention that we count from 0, i.e., we have $x = x[0]\ldots x[|x|-1]$. We use Iverson brackets $[\cdot]$ to derive bit values from Boolean conditions: For a condition C we have $[C] = 1$ if C holds; otherwise we have $[C] = 0$.

We use code-based notation for probability and security experiments. We write $\leftarrow$ for the assignment operator (that assigns a right-hand-side value to a left-hand-side variable). If S, S' are sets, we write $S \xleftarrow{\cup} S'$ shorthand for $S \leftarrow S \cup S'$. If S is a finite set, then $s \leftarrow_{\$} S$ denotes choosing s uniformly at random from S. For a randomised algorithm A we write $y \leftarrow_{\$} \mathsf{A}(x_1, x_2, \ldots)$ to denote the operation of running A with inputs $x_1, x_2, \ldots$ and assigning the output to variable y. We denote a γ-biased Bernoulli trial by $B(\gamma)$, i.e., a random experiment with possible outcomes 0 or 1 such that $\Pr[b \leftarrow_{\$} B(\gamma) : b = 1] = \gamma$. The assignments $b \leftarrow_{\$} \{0,1\}$ and $b \leftarrow_{\$} B(1/2)$ are thus equivalent. We use superscript notation to indicate when an algorithm (typically an adversary) is given access to specific oracles. An experiment terminates with a 'stop with x' instruction, where value x is understood as the outcome of the experiment. We write 'win' ('lose') as shorthand for 'stop with 1' ('stop with 0'). We write 'require C', for a Boolean condition C, shorthand for 'if not C: lose'. (We use require clauses typically to abort a game when the adversary performs some disallowed action, e.g. one that would lead to a trivial win.) The ':=' operator creates a symbolic definition; for instance, the code line '$A := E$' does *not* assign the value of expression E to variable A but instead introduces symbol A as a new (in most cases abbreviating) name for E.

2.2 Symmetric Encryption

Our syntax for symmetric encryption surfaces the randomness. Formally, an encryption scheme SE consists of algorithms $\mathsf{SE.gen}, \mathsf{SE.enc}, \mathsf{SE.dec}$. Furthermore, the scheme has associated spaces $\mathcal{K}, \mathcal{R}, \mathcal{M}, \mathcal{C}$. The key generation algorithm $\mathsf{SE.gen}$ outputs a key $k \in \mathcal{K}$. The encryption algorithm $\mathsf{SE.enc}$ takes key $k \in \mathcal{K}$, randomness $r \in \mathcal{R}$ and message $m \in \mathcal{M}$, to produce ciphertext $c \in \mathcal{C}$. We write $c \leftarrow \mathsf{SE.enc}(k, m; r)$; dropping the last input is equivalent to $r \leftarrow_{\$} \mathcal{R}$. The decryption algorithm $\mathsf{SE.dec}$ takes key k and ciphertext $c \in \mathcal{C}$ to output either a message $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ to indicate rejection.

We formalise indistinguishability under chosen-ciphertext attack for a symmetric encryption scheme via the game $\mathsf{IND\text{-}CCA}$ in Figure 1 (left). For any adversary $\mathcal{A}$ we define the advantage

$$\mathsf{Adv}^{\mathsf{ind\text{-}cca}}_{\mathsf{SE}}(\mathcal{A}) := \left|\Pr\left[\mathsf{IND\text{-}CCA}^0(\mathcal{A})\right] - \Pr\left[\mathsf{IND\text{-}CCA}^1(\mathcal{A})\right]\right|$$

and say that scheme SE is indistinguishable against chosen-ciphertext attacks if $\mathsf{Adv}^{\mathsf{ind\text{-}cca}}_{\mathsf{SE}}(\mathcal{A})$ is negligibly small for all realistic $\mathcal{A}$.

2.3 Public-Key Encryption Schemes

A PKE scheme $\mathsf{PKE} = (\mathsf{PKE.gen}, \mathsf{PKE.enc}, \mathsf{PKE.dec})$ consists of a triple 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 $\mathsf{PKE.gen}$ 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 $\mathsf{PKE.enc}$ takes a public key pk, randomness $r \in \mathcal{R}$ and a message $m \in \mathcal{M}$ to produce a ciphertext $c \in \mathcal{C}$. We write $c \leftarrow \mathsf{PKE.enc}(pk, m; r)$; dropping the last input is equivalent to $r \leftarrow_\$ \mathcal{R}$. Finally, the decryption algorithm $\mathsf{PKE.dec}$ takes a private key sk and a ciphertext $c \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ to indicate rejection. The correctness requirement is that for $(pk, sk) \leftarrow_\$ \mathsf{gen}$, $m \in \mathcal{M}$, $c \leftarrow \mathsf{PKE.enc}(pk, m)$ and $m' \leftarrow \mathsf{PKE.dec}(sk, c)$ the probability that $m' \neq m$ is upper-bounded by δ, where the probability is over all coins involved.

We formalise the indistinguishability under chosen-plaintext attack of a PKE scheme via the game $\mathsf{IND\text{-}CPA}$ in Figure 1 (centre). For any adversary $\mathcal{A}$ we define the advantage

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{PKE}}(\mathcal{A}) := |\Pr[\mathsf{IND\text{-}CPA}^0(\mathcal{A})] - \Pr[\mathsf{IND\text{-}CPA}^1(\mathcal{A})]|$$

and say that scheme PKE is indistinguishable against chosen-plaintext attacks if $\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathsf{PKE}}(\mathcal{A})$ is negligibly small for all realistic $\mathcal{A}$.

Game $\mathsf{IND\text{-}CCA}^b(\mathcal{A})$	**Game** $\mathsf{IND\text{-}CPA}^b(\mathcal{A})$	**Game** $\mathrm{sigUF}(\mathcal{A})$
00 $C \leftarrow \emptyset$	00 $C \leftarrow \emptyset$	00 $(pk, sk) \leftarrow_\$ \mathsf{DS.gen}$
01 $k \leftarrow_\$ \mathsf{SE.gen}$	01 $(pk, sk) \leftarrow \mathsf{PKE.gen}$	01 $C \leftarrow \emptyset$
02 $b' \leftarrow \mathcal{A}^{\mathrm{Enc},\mathrm{Dec}}$	02 $b' \leftarrow \mathcal{A}^{\mathrm{Enc}}(pk)$	02 $\mathcal{A}^{\mathrm{Sign},\mathrm{Vfy}}$
03 stop with b'	03 stop with b'	03 lose
Oracle $\mathrm{Enc}(m^0, m^1)$	**Oracle** $\mathrm{Enc}(m^0, m^1)$	**Oracle** $\mathrm{Sign}(m)$
04 $c \leftarrow \mathsf{SE.enc}(k, m^b)$	04 $c \leftarrow \mathsf{PKE.enc}(pk, m^b)$	04 $\mathsf{s} \leftarrow \mathsf{DS.Sign}(pk, m)$
05 $C \xleftarrow{\cup} \{c\}$	05 $C \xleftarrow{\cup} \{c\}$	05 $C \xleftarrow{\cup} \{(m, \mathsf{s})\}$
06 return c	06 return c	06 return (m, s)
Oracle $\mathrm{Dec}(c)$		**Oracle** $\mathrm{Vfy}(m, \mathsf{s})$
07 require $c \notin C$		07 $m \leftarrow \mathsf{DS.vfy}(sk, m, \mathsf{s})$
08 $m \leftarrow \mathsf{SE.dec}(k, c)$		08 if $[m \neq \perp] \wedge [(m, \mathsf{s}) \notin C]$:
09 return m		09 win
		10 return m

Fig. 1. Left: Game modelling indistinguishability under chosen-ciphertext attacks ($\mathsf{IND\text{-}CCA}$) for a symmetric encryption scheme SE. **Centre:** Game modelling indistinguishability under chosen-plaintext attacks ($\mathsf{IND\text{-}CPA}$) for a public-key encryption scheme PKE. **Right:** Game modelling the unforgeability (sigUF) of a digital signature scheme DS.

2.4 Digital Signature Schemes

Formally, a signature scheme DS consists of algorithms $\mathsf{DS.gen}, \mathsf{DS.Sign}, \mathsf{DS.vfy}$ and associated spaces $\mathcal{K}_\mathsf{S}, \mathcal{K}_\mathsf{R}, \mathcal{M}, \mathcal{S}$. The key generation algorithm $\mathsf{DS.gen}$ outputs a key pair $(pk, sk) \in \mathcal{K}_\mathsf{R} \times \mathcal{K}_\mathsf{S}$. The signing algorithm $\mathsf{DS.Sign}$ takes a signing key $sk \in \mathcal{K}_\mathsf{S}$ and a message $m \in \mathcal{M}$, and returns a message, signature pair $(m, \mathsf{s}) \in \mathcal{M} \times \mathcal{S}$. The verification algorithm $\mathsf{DS.vfy}$ takes a key $pk \in \mathcal{K}_\mathsf{R}$, a message $m \in \mathcal{M}$, and a signature $\mathsf{s} \in \mathcal{S}$, and returns either the message m (indicating that the signature is accepted) or the special symbol $\perp$ to indicate rejection.[4] For correctness we require that for all (pk, sk) output by $\mathsf{DS.gen}$ and all messages $m \in \mathcal{M}$, we have

[4] It is more common to consider the output of a verification algorithm to be a bit representing acceptance or rejection; this can be obtained from our syntax by evaluating $[\mathsf{DS.Sign}(k, m, \mathsf{s}) = m]$.

$\Pr[\mathsf{DS.vfy}(pk, m, \mathsf{DS.Sign}(sk, m)) = m] = 1$. We formalise the unforgeability of a signature scheme via the game sigUF in Figure 1 (right). For any adversary $\mathcal{A}$ we define the advantage $\mathsf{Adv}^{\mathrm{siguf}}_{\mathsf{DS}}(\mathcal{A}) := \Pr[\mathrm{sigUF}(\mathcal{A})]$ and say that the scheme DS is unforgeable if $\mathsf{Adv}^{\mathrm{siguf}}_{\mathsf{DS}}(\mathcal{A})$ is negligibly small for all realistic $\mathcal{A}$.

3 Notions of Subversion Attacks

We consider subversions of cryptographic schemes implementing encrypted communication between two parties. Abstractly, we consider a cryptographic 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 $n \in \mathbb{N}_{>0}$ algorithms on the sender side and an algorithm on the receiver side. As we consider encryption schemes, we let $\Pi.\mathsf{S}^{(0)}$ represent encryption and write $\Pi.\mathsf{S} := \Pi.\mathsf{S}^{(0)}$; our generic syntax allows for the inclusion of randomness generators as well as applying to schemes such as DPKE and FHE which require additional sender algorithms. The receiver algorithm $\Pi.\mathsf{R}$ represents decryption. Our abstract treatment allows us to capture both PKE schemes (Section 2.3) and symmetric encryption (Section 2.2), where we set $k_{\mathsf{S}} = k_{\mathsf{R}}$. Our definitions follow those of Armour and Poettering [3], which subtly extend prior definitions [7,6,18] to include subverted receivers.

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. A shortcut notation for this syntax is

$$\Pi.\mathsf{gen} \to \mathcal{K}_{\mathsf{S}} \times \mathcal{K}_{\mathsf{R}}, \quad \mathcal{K}_{\mathsf{S}} \times \mathcal{X}^{(i)} \to \Pi.\mathsf{S}^{(i)} \to \mathcal{Y}^{(i)}, \quad \text{and} \quad \mathcal{K}_{\mathsf{R}} \times \mathcal{Y} \to \Pi.\mathsf{R} \to \mathcal{X}'.$$

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.

A scheme Π is said to be δ-correct if for all $(k_{\mathsf{S}}, k_{\mathsf{R}}) \leftarrow \Pi.\mathsf{gen}$ and $x \in \mathcal{X}$ and $y \leftarrow \Pi.\mathsf{S}(k_{\mathsf{S}}, x)$ and $x' \leftarrow \Pi.\mathsf{R}(k_{\mathsf{R}}, y)$ we have $\Pr[x' \neq x] \le \delta$, where the probability is over all random coins involved. In the case that $\delta = 0$, the scheme is said to be perfectly correct.

In the following, we give formal definitions for subversion of the sender algorithm,[5] together with the notion of *undetectability* (UD). 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. (The same technique is used in most prior work, starting with [7].) In what follows we denote the subversion key space with $\mathcal{I}_{\mathsf{S}}$. In this section we also specify, by introducing notions of subliminal message recoverability, how we measure the quality of a subversion from the point of view of the subverting adversary (who is assumed to know the subversion keys).

3.1 Undetectable Subversion

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

$$\mathcal{K}_{\mathsf{S}} \times \mathcal{X} \to \Pi.\mathsf{S}_i \to \mathcal{Y}.$$

That is, for all $i \in \mathcal{I}_{\mathsf{S}}$ the algorithm $\Pi.\mathsf{S}_i$ can syntactically replace the algorithm $\Pi.\mathsf{S}$.

[5] A more general syntax would allow for the possibility that all of the scheme's algorithms are subverted; however, as we consider subverted encryption in this work we do not consider this more general case. The definitions are analogous and can easily be generated.

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 Figure 2 (centre). For any adversary $\mathcal{A}$ we define the advantage

$$\mathsf{Adv}^{\mathrm{uds}}_{\Pi}(\mathcal{A}) := |\Pr[\mathrm{UDS}^1(\mathcal{A})] - \Pr[\mathrm{UDS}^0(\mathcal{A})]|$$

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

3.2 Subliminal Information Exfiltration

We observed above that if the sender component $\Pi.\mathsf{S}$ of a cryptographic scheme Π is undetectably subverted, with uniformly chosen index i_{S} that remains unknown to the participants, then all security guarantees are preserved from the original scheme. This may be different if i_{S} is known to an attacking party, and indeed we assume that mass-surveillance attackers leverage such knowledge to conduct attacks.

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 Figure 2 (left), 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 (reflecting the fact that, in the contexts we consider, inputs to $\Pi.\mathsf{S}$ typically represent messages) that produces the inputs to $\Pi.\mathsf{S}$ used throughout the game. The syntax of this message sampler is

$$\Sigma \times A \to \mathrm{MS} \to \Sigma \times \mathcal{X} \times B, \quad (\sigma, \alpha) \mapsto \mathrm{MS}(\sigma, \alpha) = (\sigma', x, \beta),$$

where $\sigma, \sigma' \in \Sigma$ are old and updated state, input $\alpha \in A$ models the influence that the adversary may have on message generation, and output $\beta \in B$ models side-channel outputs. In Figure 2 we write $\diamond$ for the initial state. Note that while we formalise the inputs α and the outputs β for generality (so that our models cover most real-world applications), our subversion attacks are independent of them. For any message sampler MS and adversary $\mathcal{A}$ we define the advantage

$$\mathsf{Adv}^{\mathsf{mr}}_{\Pi,\mathrm{MS}}(\mathcal{A}) := \Pr[\mathsf{MR}(\mathcal{A})].$$

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 $\mathsf{Adv}^{\mathsf{mr}}_{\Pi,\mathrm{MS}}(\mathcal{A})$ reaches a considerable value (e.g., 0.1).[6]

3.3 Generic Method: Rejection Sampling

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 that encodes the subliminal message. We define a subversion of the encryption algorithm $\Pi.\mathsf{S}$ of a scheme Π in Figure 2 (right, top). It 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 write $\Pi.\mathsf{S}_i$ for the subverted algorithm. We give a corresponding message recovery adversary in Figure 2 (right, bottom).

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). We note that this technique embeds a message of length ℓ_μ in each ciphertext; in later sections we use this idea to exfiltrate a message

[6] 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.

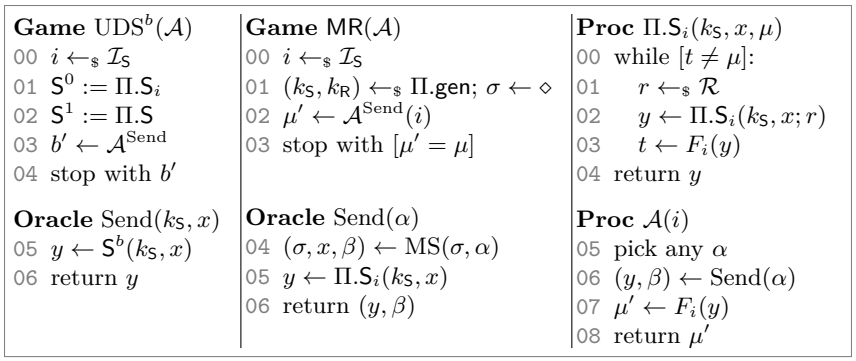

Fig. 2. Left: Game UDS modelling sender subversion undetectability for a scheme Π. **Centre:** Game MR modelling key recoverability for passive adversaries. **Right:** Rejection sampling subversion $\Pi.\mathsf{S}_i$ of encryption algorithm $\Pi.\mathsf{S}$ and corresponding message recovering adversary $\mathcal{A}$, as in Section 3.2. The adversary need not have any influence over messages (modelled by α; see the discussion at Section 3.2).

that is derived from the plaintext being encrypted. More generally, each subliminal message μ could be the fragment of a larger message $\tilde{\mu}$ (e.g. representing the secret key, as is the approach in prior work targeting symmetric encryption). It is straightforward to see how this would work for a stateful algorithm (simply send the bits in order); for a stateless algorithm, Bellare et al. [6] show that if individual ciphertexts embed messages of length ℓ_μ then it is possible to exfiltrate a string $\tilde{\mu}$ of length 2^{ℓ_μ} by letting each individual μ encode the ${\ell_\mu}^{\text{th}}$ bit of $\tilde{\mu}$.

3.4 Defending Against Subversion Attacks

Achieving security against adversaries mounting ASAs is difficult, and essentially reduces to assuming trust in particular components or architectures. The three main theoretical approaches to preventing or mitigating against ASAs in the literature are reverse firewalls, self-guarding protocols and watchdogs. Another approach is given by Bellare and Hoang [5] who discuss deterministic PKE schemes that defend against the subversion of random number generators.

Cryptographic reverse firewalls [28,20,27,39,11] represent an architecture to counter ASAs against asymmetric cryptography via trusted code in network perimeter filters. At a high level, the approach is for a trusted third party to re-randomise ciphertexts before transmission over a public network to destroy any subliminal messages. Fischlin and Mazaheri show how to construct 'self-guarding' ASA-resistant (asymmetric) encryption and signature algorithms given initial access to a trusted base scheme [23]. Their approach uses trusted samples to essentially perform re-randomisation of ciphertexts.

In a series of work, Russell, Tang, Yung and Zhou [31,32,33,34] study ASAs on one-way functions, trapdoor one-way functions and key generation as well as defending randomised algorithms against ASAs using so-called watchdogs. The watchdog model considers splitting a primitive into constituent algorithms that are run as subroutines by a trusted 'amalgamation' layer. This allows the constituent algorithms to be individually checked and sanitised, in a variety of different assumptions (e.g. on- or offline, black- or whitebox access). Combiners are often used to provide subversion resilience, particularly in the watchdog model. A combiner [24,30] essentially combines the output from different algorithms (or runs of the same algorithm) in such a way as to produce secure (in this case, unsubverted) combined output as long as any one of the underlying outputs is secure. Bemman, Chen and Jager [8] show how to construct a subversion-resilient KEM, using a variant of a combiner and a subversion resilient randomness generator. Their construction considers Russell et al.'s watchdog from a practical perspective, meaning an offline watchdog that runs in linear time. Another line of work, [21,4,19], examined backdoored hash functions, showing how to immunise hash functions against subversion.

4 Case Study: Symmetric Cryptography

In this section we describe an illustrative case study that serves to introduce the concepts of deniability and subversion; in order to highlight the intuition behind our ideas, our discussion proceeds rather informally. We show how a subversion attack against symmetric encryption schemes can undermine the deniability that the scheme provides. Later, in Section 5, we discuss deniable public-key encryption, where the notions of deniability are more subtle and require a more formal treatment.

Symmetric encryption schemes are intuitively deniable, in the following sense: If Alice and Bob share a secret key, then any ciphertext could have been created by either party. If we consider messages in the direction from Alice to Bob, this means that Bob is unable to present an adversary with a convincing proof that Alice sent a particular message (by revealing a key, message and ciphertext that he claims were sent by Alice). This means that it is ineffective for an adversary to coerce Bob to reveal Alice's messages.[7] Further, it also means that Bob is unable to convincingly 'frame' Alice for messages she didn't send. The inherent deniability provided by symmetric encryption is usually considered in the context of non-repudiation, where it is regarded as a weakness. Non-repudiation can be achieved via digital signatures, an asymmetric primitive that allows a signer to create signatures for messages such that only the signer could have created the signature.

Deniable 'shared-key' encryption was considered by Canetti et al. [13,12], who considered a stronger notion of deniability analogous to public-key deniability – that is, a deniable symmetric scheme that allows a sender to later claim that a different key and message were used to encrypt than those actually used. Canetti et al. give the example of a one-time pad as a scheme that meets this deniability notion. Another construction is to encrypt a tuple of ℓ messages with ℓ keys (the i^{th} key is used to encrypt the i^{th} message). The symmetric key shared by Alice and Bob is the tuple of ℓ encryption keys together with an index referring to the intended message. Later, when coerced, Alice can claim that a different index was used during encryption.

We consider a scenario where a symmetric encryption scheme is used for its inherent deniability property.[8] In the remainder of this section, we show that a subversion adversary who can subvert the scheme's encryption algorithm is able to undermine deniability by introducing a subliminal channel in ciphertexts containing a digital signature. This means that if Bob reveals Alice's messages, her ability to deny that she sent the messages is undermined, as an adversary who observes ciphertexts is able to obtain the commitment to the underlying messages contained within the subliminal channel. Furthermore, the fact that ciphertexts commit to underlying messages is undetectable according to the undetectability notions of Section 3.1 – that is, any detector with black-box access to the subverted scheme will be unable to determine whether the scheme is subverted.

We first recall two methods to implant a subliminal channel into ciphertexts generated using symmetric encryption: IV replacement and biased ciphertexts. We then go on to informally describe how to subvert deniability of symmetric encryption schemes using a subliminal channel, which serves as a useful case study for our results in Section 5.

Rejection Sampling Rejection sampling, as discussed in Section 3.3, allows a subliminal channel to be implanted in ciphertexts. As noted before, in practice the subliminal channel's bandwidth needs to be small (one or two bits) in order to ensure that the algorithm is not prohibitively slow.

IV replacement Following [7], we can also consider IV replacement. Consider a randomised stateless scheme $\mathsf{SE} = (\mathsf{SE.gen}, \mathsf{SE.enc}, \mathsf{SE.dec})$. We write $c \leftarrow \mathsf{SE.enc}(k, m; IV)$ to highlight the fact that we surface the randomness input IV (for initial vector) to the encryption algorithm. Such a scheme is said to surface

[7] In theory, this means there is little point for an adversary to coerce Bob to reveal messages from Alice, as no trust can be placed in the testimony. However, 'real world' deniability is a subtle concept and cannot be fully reduced to formal notions; for example, a coercive adversary may require that messages themselves are plausible (do not contradict other testimony or alibis). A particularly unscrupulous adversary may simply victimise anyone who has had correspondence with Alice.

[8] Our discussion and results are easily adapted to the stronger 'deniable shared-key' notion of Canetti et al.

its IV if there is an efficient algorithm χ such that $\chi(\mathsf{SE.enc}(k, m; IV)) = IV$ for all k, m, IV. The condition says that χ can recover the IV from the ciphertext. A simple example of a scheme that surfaces its IV is CBC$, namely CBC mode with random IV. Another example is CTR$, counter mode with random starting point.

4.1 Subverting Deniability of Symmetric Encryption

We assume that the adversary subverts the encryption algorithm $\mathsf{SE.enc}$ of a symmetric encryption scheme so that the ciphertexts contain a subliminal channel, either using IV replacement or rejection sampling. Rather than using the channel to exfiltrate the secret key, as is the approach in other settings,[9] we let the adversary transmit a commitment to the underlying message in the form of a digital signature s. Alice generates ciphertext $c \leftarrow \mathsf{SE.enc}_i(k, m)$ which encrypts a message m under key k, using the subverted encryption algorithm $\mathsf{SE.enc}_i$ so that c encodes the signature s. At a later point in time, the adversary can coerce Bob to reveal ciphertext, key, message c^*, k^*, m^* and can then compare the message m^* to the signature encoded in the ciphertext c^*. We note that for an adversary, knowing whether or not Bob is lying about the message that he sent is sufficient to conclude that Alice and Bob exchanged illicit messages.

In more detail, assume that the adversary has subverted the encryption algorithm $\mathsf{SE.enc}$ so that ciphertexts encode ℓ-bits of subliminal information. As per the discussion at Section 3, we assume that the subverted algorithm has an embedded subversion key $i \in \mathcal{I}_{\mathsf{S}}$ known to the adversary. For notational convenience, we denote the embedded subversion key by sk_i (as it represents the signing key of a digital signature scheme) and give the adversary the corresponding verification key pk_i.

On input a message m, the subverted encryption algorithm $\mathsf{SE.enc}_i$ first calculates an ℓ-bit digital signature $\mathsf{s}_\ell \leftarrow \mathsf{DS.Sign}_\ell(sk_i, m)$.[10] Then, using the subliminal channel, the signature s_ℓ is encoded into a valid ciphertext c. An adversary who is given the ciphertext c and knows the subversion key i can recover the signature s_ℓ and check (using the corresponding verification key pk_i) that it verifies against the message m^* that Bob claims was encrypted.

Success of the Subversion We first note that the distribution of subverted ciphertexts is indistinguishable from the distribution of unsubverted ciphertexts, assuming that the digital signature scheme outputs signatures whose distribution is (computationally) indistinguishable from random. In both cases (real or subverted), a distinguisher playing the subversion detection game UDS observes ciphertexts that are indistinguishable from random. This means that a detector with black box access to the subverted encryption algorithm is unable to distinguish $\mathsf{SE.enc}$ from $\mathsf{SE.enc}_i$ with any meaningful probability – that is, the attack is undetectable according to the notion in Section 3.1.

Furthermore, the attack potentially allows the adversary to tell whether a particular message corresponds to a ciphertext or not with some (non-negligible) probability. Applying Kerckhoffs' principle, we assume that the communicating parties (Alice and Bob) know that the encryption is subverted, but do not have access to the secret signing key sk_i. In effect, we assume that Alice (and Bob) have black-box access to the subverted encryption algorithm.[11] This means that Alice and Bob have access to an oracle that on input a message m returns a signature $\mathsf{DS.Sign}_\ell(sk_i, m)$ – meaning that Alice and Bob play the role of adversary in an

[9] While leaking the secret key would allow an adversary to compromise users, we are looking ahead to dPKE where the user's key is public and thus pointless to leak.

[10] We denote a digital signature scheme that outputs ℓ-bit signatures with $\mathsf{DS.Sign}_\ell$. As ℓ is typically quite small, this notion is a useful thought experiment. As we are not aware of any practical signature schemes with short tags, an alternative would be to use a MAC scheme to provide commitments to underlying messages in the form of MAC tags. As MAC tags are deterministic, it is easy to obtain short tags by truncating to ℓ bits. Note that we assume that Alice and Bob do not have access to the secret signing key i, by Kerckhoffs' principle, so that this is meaningful. Bob should be unable to forge MAC tags without knowledge of i.

[11] This assumption is a common approach in work on ASAs, e.g. [7,6]. The embedded subversion key may be obfuscated in code or stored in a trusted execution environment that a user is unable to tamper with. Using techniques from malware [25], this is a plausible outcome for an adversary.

unforgeability game UF, as discussed in Section 2.4. When Bob is coerced by the adversary, in order to be convincing he will need to produce c^*, k^*, m^* such that c^* encodes s_ℓ with $\mathsf{DS.vfy}(pk_i, \mathsf{s}_\ell, m^*) \neq \perp$. Informally, if the signature scheme is secure then Bob's advantage in this task is negligible.

We thus conclude that the subverted encryption scheme is no longer inherently deniable, and in fact the deniability of the subverted scheme reduces to the security of the signature scheme that the subverted algorithm runs as a subroutine. This security is a function of the length of signatures, expressed above as ℓ.

4.2 Discussion

IV replacement allows for $|IV|$ bits of information – commonly 128, if AES is the block cipher used – to be encoded into ciphertexts, which would make it unrealistic for Bob to deny messages (successfully evade the subverted deniability). This case is less practically relevant, as IV surfacing schemes are not widely used, but allows us to conclude that subverting deniability is a meaningful concept. We note that the rejection sampling method allows only a few bits to be implanted into the subliminal channel, which means that signatures are not long enough to be unforgeable by Bob. This means that the subversion is unsuccessful from the point of view of an adversary. Of course, this assumes that Alice and Bob are aware of the subversion and actively craft convincing c^*, k^*, m^* such that the encoded signature verifies over m^*. We note that in practice, it may be the case that the subversion goes unnoticed by Alice and Bob – and for an unscrupulous adversary this may be sufficient to undermine deniability in practice.

To conclude, in this section we discussed how the deniability of symmetric encryption schemes can be undermined if the algorithms are subverted. The subversion techniques and deniability notions translate loosely onto deniable public-key encryption, which we discuss in the next section. For deniable PKE, an adversary can used subliminal channels within ciphertexts or within the randomness revealed during coercion, so that subverting deniability becomes more feasible – and highly relevant, considering that deniability is an explicit design goal.

5 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 work we focus on non-interactive sender deniable public-key encryption, as introduced by Canetti et al. (CDNO) [12], who showed that a sender-deniable scheme can be used to construct receiver deniable (and thus bi-deniable) schemes. Other notions of deniability include weak (or 'multi-distributional') deniability in which a sender uses an alternative ('fake') encryption algorithm to encrypt deniable messages – when coerced, they claim to have run the regular algorithm. Canetti et al. describe such a scheme in [12]; later O'Neill et al. [29] proposed a non-interactive encryption scheme with negligible deniability simulatable encryption. Another line of work uses Indistinguishability Obfuscation (iO) to achieve deniable encryption: Sahai and Water's sender deniable scheme [35] and Canetti Park and Poburinnaya's bi-deniable interactive scheme [14]. However, the current state of iO means that these result serve more as a theoretical feasibility result. De Caro, Iovino and O'Neill [17] studied the notion of receiver deniable functional encryption, but instantiating their constructions required fully fledged functional encryption, which in turn is known to imply iO.

To date, no practical deniable schemes has been proposed. Either deniability is not practically achievable, as in the case of the CDNO Parity Scheme whose ciphertexts grow inversely proportional to the deniability probability, or else the construction requires strong assumptions such as iO or functional encryption. Recent work by Agrawal et al. [1] is promising in this regard, as their construction for deniable fully homomorphic

encryption (FHE) provides compact ciphertexts and is based on the security of Learning with Errors. Nevertheless, their construction requires a running time that is inversely proportional to detection probability. In the absence of practical schemes, we demonstrate the feasibility of our ASA targeting deniable encryption schemes (Section 6) by focussing on the illustrative case study of the CDNO 'Parity Scheme'.

The remainder of this section sets the scene for our attack in Section 6; we first recap the formal definition of a deniable PKE scheme in Section 5.1 before describing the CDNO Parity Scheme in Section 5.2.

5.1 Definition of Deniable PKE Scheme

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 special symbol $\perp \notin \mathcal{M}$ to indicate rejection.
- Finally, the faking algorithm $\mathsf{DE.Fake}$ takes a public key pk, a pair of messages and randomness m, r as well as a 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 $(\mathsf{DE.gen}, \mathsf{DE.enc}, \mathsf{DE.dec})$ satisfy the standard notions of correctness and IND-CPA security properties of public-key encryption, as in Section 2.3. We formalise the deniability of the scheme via the game INDEXP in Figure 3. 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

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

We say that scheme DE is deniable if $\mathsf{Adv}^{\mathsf{indexp}}_{\mathsf{DE}}(\mathcal{A})$ is negligibly small for all realistic $\mathcal{A}$.

Note that a scheme cannot simultaneously satisfy perfect correctness and deniability, so negligible decryption error in correctness is inherent.

Game $\mathsf{INDEXP}^b(\mathcal{A})$	**Game** $\mathsf{subINDEXP}^b(\mathcal{A})$
00 $(dpk, dsk) \leftarrow \mathsf{DE.gen}$	00 $(dpk, dsk) \leftarrow \mathsf{DE.gen}$; $i \in \mathcal{I}_\mathsf{S}$
01 $b' \leftarrow \mathcal{A}^{\mathrm{Enc}}(dpk)$	01 $b' \leftarrow \mathcal{A}^{\mathrm{Enc}}(dpk)$
02 stop with b'	02 stop with b'
Oracle $\mathrm{Exp}(m, m^*)$	**Oracle** $\mathrm{Exp}(m, m^*)$
03 $r \leftarrow_\$ \mathcal{R}$	03 $r \leftarrow_\$ \mathcal{R}$
04 $r^* \leftarrow \mathsf{DE.Fake}(dpk, m, r, m^*)$	04 $r^* \leftarrow \mathsf{DE.Fake}(dpk, m, r, m^*)$
05 if $b = 0$:	05 if $b = 0$:
06 return $(m^*, r, \mathsf{DE.enc}(dpk, m^*; r))$	06 return $(m^*, r, \mathsf{DE.enc}_i(dpk, m^*; r))$
07 else:	07 else:
08 return $(m^*, r^*, \mathsf{DE.enc}(dpk, m; r))$	08 return $(m^*, r^*, \mathsf{DE.enc}_i(dpk, m; r))$

Fig. 3. Games modelling the deniability (indistinguishability of explanation) of a deniable PKE scheme (left) and a subverted deniable PKE scheme (right).

5.2 CDNO Parity Scheme

Here we describe the sender deniable 'Parity Scheme' of Canetti et al. [12]. Informally, ciphertexts consist of a tuple of elements where each element is either chosen randomly from a set $T = \{0,1\}^t$ or a so-called 'translucent set' S_t, where S satisfies the following properties:

- $S_t \subset T$ and $|S_t| \leq 2^{t-k}$, for sufficiently large k.
- It is easy to generate random elements $x \in S_t$.
- Given $x \in T$ and trapdoor information d_t, it is easy to check whether $x \in S_t$.
- Without d_t it is computationally infeasible to decide whether $x \in S_t$.

For specificity, we consider the construction of translucent sets given in [12] based on a trapdoor permutation $f\colon \{0,1\}^s \to \{0,1\}^s$ and its hard-core predicate $B\colon \{0,1\}^s \to \{0,1\}$. Let $t = s + k$. Represent each $x \in T$ as $x = x_0 \parallel b_1 \parallel b_2 \parallel \ldots \parallel b_k$, where $x_0 \in \{0,1\}^s$ is followed by k bits. Then the translucent set is defined as:

$$S = \left\{ x = x_0 \parallel b_1 \parallel b_2 \parallel \ldots \parallel b_k \in \{0,1\}^{s+k} | (\forall i \leq k) B(f^{-i}(x_0)) = b_i \right\}.$$

The trapdoor information d_t plays the role of a private key.

We give a description in pseudo-code of the encryption algorithm $\mathsf{PS.enc}$ in Figure 4. On input a bit value b, the encryption algorithm first chooses a random number $0 < \ell \leq n$ with parity b in line 00. Next, ℓ elements in S are generated in lines 02 to 06. Lastly, before outputting the ciphertext in line 09, $n - \ell$ elements in T are generated in lines 07 and 08. We refer the reader to [12,13] for full details of the scheme, including decryption and faking algorithms as well as proofs of the security and deniability of the scheme. In particular, it is shown that the parity scheme is a $4/n$-sender deniable encryption scheme, which means that the probability of a successful attack of a coercer vanishes linearly in the security parameter n.

Proc $\mathsf{PS.enc}(pk, m)$	**Proc** $\mathsf{PS.enc}_i(pk, m)$
	00 $\mathsf{s} \leftarrow \mathsf{DS.Sign}_{(n+1)s}(sk_i, m)$.
00 while $\ell \mod 2 \neq b$:	01 while $\ell \mod 2 \neq b$:
01 $\quad \ell \leftarrow_\$ [0\,..\,n+1]$	02 $\quad \ell \leftarrow_\$ [0\,..\,n+1]$
02 for $i \in [0\,..\,\ell]$:	03 for $i \in [0\,..\,\ell]$:
03 $\quad x_0^{(i)} \leftarrow_\$ \{0,1\}^s$	04 $\quad x_0^{(i)} \leftarrow_\$ \mathsf{s}[is : (i+1)s]$
04 $\quad$ for $j \in [0\,..\,k]$:	05 $\quad$ for $j \in [0\,..\,k]$:
05 $\quad\quad b_j^{(i)} \leftarrow B(f^{-j}(x_0^{(i)}))$	06 $\quad\quad b_j^{(i)} \leftarrow B(f^{-j}(x_0^{(i)}))$.
06 $\quad x^{(i)} \leftarrow x_0^{(i)} \parallel b_0^{(i)} \parallel \ldots \parallel b_k^{(i)}$	07 $\quad x^{(i)} \leftarrow x_0^{(i)} \parallel b_0^{(i)} \parallel \ldots \parallel b_k^{(i)}$
07 for $i \in [\ell\,..\,n+1]$:	08 for $i \in [\ell\,..\,n+1]$:
08 $\quad x^{(i)} \leftarrow_\$ \{0,1\}^t$	09 $\quad x_0^{(i)} \leftarrow_\$ \mathsf{s}[is : (i+1)s]$
	10 $\quad x_1^{(i)} \leftarrow_\$ \{0,1\}^k$
	11 $\quad x^{(i)} \leftarrow x_0^{(i)} \parallel x_1^{(i)}$
09 return $c = (x^{(0)}, x^{(1)}, \ldots, x^{(n)})$	12 return $c = (x^{(0)}, x^{(1)}, \ldots, x^{(n)})$

Fig. 4. Left: CDNO Parity Scheme encryption algorithm $\mathsf{PS.enc}$. **Right:** Subverted encryption algorithm $\mathsf{PS.enc}_i$.

6 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. Applying the definition, an adversary playing the INDEXP game

(Figure 3) is able to distinguish between real and fake cases and win the game with non-negligible probability. 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 follow the approach given in the symmetric encryption case study discussed in Section 4. This approach uses subversion techniques to embed a subliminal channel in ciphertexts that transmits a commitment to the message – we assume that the commitment takes the form of a digital signature. The generic rejection sampling technique is unable to provide enough bandwidth to transmit sufficiently long signatures, 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. As a particular illustrative example, we demonstrate this approach in the case of the CDNO Parity Scheme (Section 5.2). As the scheme surfaces its randomness, the ciphertexts reveal the underlying randomness making it particularly straight forward to embed a 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 deniable PKE 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.[12]

Formal Definition of Subverted Deniable PKE We note that a deniable PKE scheme satisfies the generic syntax introduced above in Section 3, 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 Sections 3.1 and 3.2. We obtain the game $\mathsf{subINDEXP}$ given in Figure 3 (right), modelling the adversary's ability to compromise the deniability property of a subverted scheme. The adversary's goal is to win the subverted deniability game $\mathsf{subINDEXP}$.

6.1 Case Study: Subverting CDNO Parity Scheme

We now discuss an ASA against the CDNO parity scheme, described in Section 5.2. As we note above, this is a useful case study to demonstrate the feasibility of an ASA targeting deniability. We give details of our subversion attack in Figure 4 (right). At a high level, we subvert randomness so that they encode a commitment to the original message in the form of a digital signature. Our technique is analogous to the IV replacement attack described in Section 4.

On input a public key and bit message m, the subverted algorithm first generates a signature of length $(n+1)s$ bits over the plaintext m using its secret signing key sk_i (line 00). This signature will be embedded into the 'random' elements that constitute the ciphertext in chunks. When generating elements from S

[12] 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.

in lines 03 to 07, the subverted encryption algorithm uses chunks of the signature (line 04) rather than sampling a random element. lines 05 to 07 ensures that we are generating elements in S. When generating elements from T in lines 08 to 11, the subverted encryption algorithm sets the first s bits to be a chunk of the signature, then samples the remaining bits at random. Finally, the subverted algorithm outputs the ciphertext in line 09.

We note that ciphertexts are well-formed and the subverted ciphertext c encrypts the message m as intended. To see this, the recipient with knowledge of the trapdoor d_t will be able to identify which elements are in S.

Success of the Subversion We first note that the distribution of subverted ciphertexts is indistinguishable from the distribution of unsubverted ciphertexts, assuming that the digital signature scheme outputs signatures whose distribution is (computationally) indistinguishable from random. In both cases (real or subverted), a distinguisher without knowledge of the trapdoor d_t playing the subversion detection game UDS observes ciphertexts that are indistinguishable from random. This means that a detector with black box access to the subverted encryption algorithm is unable to distinguish $\mathsf{PS.enc}$ from $\mathsf{PS.enc}_i$ with any meaningful probability – that is, the attack is undetectable according to the notion in Section 3.1.

An adversary who is given the ciphertext c (or indeed the randomness r) and knows the subversion key i can recover the signature s and check (using the verification key pk_i) that it verifies against the message m^* that Bob claims was encrypted. More formally, a subversion adversary playing the $\mathsf{subINDEXP}$ game from Figure 3 (right) is able to distinguish between $(m^*, r, \mathsf{DE.enc}_i(dpk, m^*; r))$ and $(m^*, r^*, \mathsf{DE.enc}_i(dpk, m; r))$ by recovering s from the randomness and testing whether $\mathsf{DS.vfy}(pk_i, m^*, \mathsf{s})$ verifies. We note that as the randomness encodes the digital signature, recovering the signature is independent of the ciphertext and this method applies generically even to encryption schemes that do not surface their randomness.

Lastly, we note that even if Alice is aware that her encryption algorithm is subverted as long as she does not have access to the secret signing key sk_i she is unable to forge a signature which would allow her to claim she sent a fake message. We thus conclude that the subverted scheme is no longer deniable, and in fact the deniability of the subverted scheme reduces to the security of the signature scheme that the subverted algorithm runs as a subroutine. Thus security is a function of the length of signature – in the example of the parity scheme, this is sufficiently large to a meaningful success probability to the adversary.

6.2 Subversion-Resilient Deniable PKE Schemes

Following the discussion at Section 3.4, we may apply any of the standard approaches (reverse firewalls, self-guarding protocols or watchdogs) to sanitise the scheme and prevent subliminal channels in ciphertexts. One way to achieve this generically is to simply to compose the deniable PKE scheme DE with a subversion resilient PKE scheme $\mathsf{PKE_{SR}}$ so that the output of $\mathsf{DE.enc}$ is encrypted under $\mathsf{PKE_{SR}}$ before being sent to the receiver. Particular deniable PKE constructions may allow a more efficient approach; for example, reverse firewalls apply directly to deniable FHE.

However, removing subliminal channels is not sufficient to protect against subversion, as the adversary is still able to coerce the sender to reveal randomness at some point in the future. To mitigate this, deniable PKE constructions should explicitly separate randomness generation from encryption so that $\mathsf{DE.enc}$ is deterministic, following the approach of [5].

We note that neither measure (sanitising subliminal channels, separation between randomness generation and encryption) is sufficient in isolation. Separating randomness generation from encryption does not necessarily result in ciphertexts that are free of a subliminal channel – whilst it protects against the rejection sampling method, other non-generic methods, such as [40], could potentially result in subverted ciphertexts.

7 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 claim that she is not a participant

in a particular communication [22]. However, the adversarial model and evaluation of real world protocols claiming deniability is not agreed upon; Celi and Symeonidis [15] 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 coerce users. 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.

Further Work An interesting open problem is to consider subversion attacks against deniable protocols, for example the Signal protocol [38] or deniable key agreement. At present, subverting protocols has not received a great deal of attention other than recent work by Berndt et al. [10], who considered subverting protocols including Signal with the aim of leaking a secret key.

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. Cryptology ePrint Archive, Report 2019/987 (2019), https://eprint.iacr.org/2019/987
3. Armour, M., Poettering, B.: Algorithm substitution attacks against receivers. Cryptology ePrint Archive, Report 2022/604 (2022), https://eprint.iacr.org/2022/604
4. Bauer, B., Farshim, P., Mazaheri, S.: Combiners for backdoored random oracles. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part II. Lecture Notes in Computer Science, vol. 10992, pp. 272–302. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_10
5. Bellare, M., Hoang, V.T.: Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part II. Lecture Notes in Computer Science, vol. 9057, pp. 627–656. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_21
6. 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
7. Bellare, M., Paterson, K.G., Rogaway, P.: Security of symmetric encryption against mass surveillance. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology – CRYPTO 2014, Part I. Lecture Notes in Computer Science, vol. 8616, pp. 1–19. Springer, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_1
8. Bemmann, P., Chen, R., Jager, T.: Subversion-resilient public key encryption with practical watchdogs. In: Garay, J. (ed.) PKC 2021: 24th International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 12710, pp. 627–658. Springer, Heidelberg (May 2021). https://doi.org/10.1007/978-3-030-75245-3_23
9. 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
10. Berndt, S., Wichelmann, J., Pott, C., Traving, T.H., Eisenbarth, T.: ASAP: Algorithm substitution attacks on cryptographic protocols. Cryptology ePrint Archive, Report 2020/1452 (2020), https://eprint.iacr.org/2020/1452
11. Bossuat, A., Bultel, X., Fouque, P.A., Onete, C., van der Merwe, T.: Designing reverse firewalls for the real world. In: Chen, L., Li, N., Liang, K., Schneider, S.A. (eds.) ESORICS 2020: 25th European Symposium on Research in Computer Security, Part I. Lecture Notes in Computer Science, vol. 12308, pp. 193–213. Springer, Heidelberg (Sep 2020). https://doi.org/10.1007/978-3-030-58951-6_10
12. 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

13. Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable encryption. Cryptology ePrint Archive, Report 1996/002 (1996), https://eprint.iacr.org/1996/002
14. 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
15. Celi, S., Symeonidis, I.: The current state of denial. In: HotPETS (2020)
16. 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
17. 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
18. Degabriele, J.P., Farshim, P., Poettering, B.: A more cautious approach to security against mass surveillance. In: Leander, G. (ed.) Fast Software Encryption – FSE 2015. Lecture Notes in Computer Science, vol. 9054, pp. 579–598. Springer, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-48116-5_28
19. Dodis, Y., Farshim, P., Mazaheri, S., Tessaro, S.: Towards defeating backdoored random oracles: Indifferentiability with bounded adaptivity. In: Pass, R., Pietrzak, K. (eds.) TCC 2020: 18th Theory of Cryptography Conference, Part III. Lecture Notes in Computer Science, vol. 12552, pp. 241–273. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64381-2_9
20. Dodis, Y., Mironov, I., Stephens-Davidowitz, N.: Message transmission with reverse firewalls—secure communication on corrupted machines. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 341–372. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53018-4_13
21. Fischlin, M., Janson, C., Mazaheri, S.: Backdoored hash functions: Immunizing HMAC and HKDF. In: Chong, S., Delaune, S. (eds.) CSF 2018: IEEE 31st Computer Security Foundations Symposium. pp. 105–118. IEEE Computer Society Press (2018). https://doi.org/10.1109/CSF.2018.00015
22. Fischlin, M., Mazaheri, S.: Notions of deniable message authentication. In: Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society. p. 55?64. WPES '15, Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2808138.2808143, https://doi.org/10.1145/2808138.2808143
23. Fischlin, M., Mazaheri, S.: Self-guarding cryptographic protocols against algorithm substitution attacks. In: Chong, S., Delaune, S. (eds.) CSF 2018: IEEE 31st Computer Security Foundations Symposium. pp. 76–90. IEEE Computer Society Press (2018). https://doi.org/10.1109/CSF.2018.00013
24. Giacon, F., Heuer, F., Poettering, B.: KEM combiners. In: Abdalla, M., Dahab, R. (eds.) PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 10769, pp. 190–218. Springer, Heidelberg (Mar 2018). https://doi.org/10.1007/978-3-319-76578-5_7
25. Gollmann, D.: Computer Security (3. ed.). Wiley (2011), http://eu.wiley.com/WileyCDA/WileyTitle/productCd-1118801326.html
26. 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
27. Ma, H., Zhang, R., Yang, G., Song, Z., Sun, S., Xiao, Y.: Concessive online/offline attribute based encryption with cryptographic reverse firewalls - secure and efficient fine-grained access control on corrupted machines. In: López, J., Zhou, J., Soriano, M. (eds.) ESORICS 2018: 23rd European Symposium on Research in Computer Security, Part II. Lecture Notes in Computer Science, vol. 11099, pp. 507–526. Springer, Heidelberg (Sep 2018). https://doi.org/10.1007/978-3-319-98989-1_25
28. Mironov, I., Stephens-Davidowitz, N.: Cryptographic reverse firewalls. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part II. Lecture Notes in Computer Science, vol. 9057, pp. 657–686. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_22
29. O'Neill, A., Peikert, C., Waters, B.: Bi-deniable public-key encryption. In: Rogaway, P. (ed.) Advances in Cryptology – CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 525–542. Springer, Heidelberg (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9_30
30. Poettering, B., Rösler, P.: Combiners for AEAD. IACR Transactions on Symmetric Cryptology **2020**(1), 121–143 (2020). https://doi.org/10.13154/tosc.v2020.i1.121-143
31. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Cliptography: Clipping the power of kleptographic attacks. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology – ASIACRYPT 2016, Part II. Lecture Notes in Computer Science, vol. 10032, pp. 34–64. Springer, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53890-6_2

32. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Destroying steganography via amalgamation: Kleptographically CPA secure public key encryption. Cryptology ePrint Archive, Report 2016/530 (2016), https://eprint.iacr.org/2016/530
33. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Generic semantic security against a kleptographic adversary. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017: 24th Conference on Computer and Communications Security. pp. 907–922. ACM Press (Oct / Nov 2017). https://doi.org/10.1145/3133956.3133993
34. Russell, A., Tang, Q., Yung, M., Zhou, H.S.: Correcting subverted random oracles. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part II. Lecture Notes in Computer Science, vol. 10992, pp. 241–271. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_9
35. 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
36. 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
37. Simmons, G.J.: The prisoners' problem and the subliminal channel. In: Chaum, D. (ed.) Advances in Cryptology – CRYPTO'83. pp. 51–67. Plenum Press, New York, USA (1983)
38. Vatandas, N., Gennaro, R., Ithurburn, B., Krawczyk, H.: On the cryptographic deniability of the Signal protocol. In: Conti, M., Zhou, J., Casalicchio, E., Spognardi, A. (eds.) ACNS 20: 18th International Conference on Applied Cryptography and Network Security, Part II. Lecture Notes in Computer Science, vol. 12147, pp. 188–209. Springer, Heidelberg (Oct 2020). https://doi.org/10.1007/978-3-030-57878-7_10
39. Wang, Y., Chen, R., Huang, X., Wang, B.: Secure anonymous communication on corrupted machines with reverse firewalls. IEEE Transactions on Dependable and Secure Computing pp. 1–1 (2021). https://doi.org/10.1109/TDSC.2021.3107463
40. Yang, Z., Chen, R., Li, C., Qu, L., Yang, G.: On the Security of LWE Cryptosystem against Subversion Attacks. The Computer Journal **63**(4), 495–507 (09 2019). https://doi.org/10.1093/comjnl/bxz084, https://doi.org/10.1093/comjnl/bxz084
41. Young, A., Yung, M.: The dark side of "black-box" cryptography, or: Should we trust capstone? In: Koblitz, N. (ed.) Advances in Cryptology – CRYPTO'96. Lecture Notes in Computer Science, vol. 1109, pp. 89–103. Springer, Heidelberg (Aug 1996). https://doi.org/10.1007/3-540-68697-5_8
42. Young, A., Yung, M.: Kleptography: Using cryptography against cryptography. In: Fumy, W. (ed.) Advances in Cryptology – EUROCRYPT'97. Lecture Notes in Computer Science, vol. 1233, pp. 62–74. Springer, Heidelberg (May 1997). https://doi.org/10.1007/3-540-69053-0_6