# Subverting Decryption in AEAD[⋆]

Marcel Armour[1](✉) and Bertram Poettering[2]

[1] Information Security Group, Royal Holloway University of London
[2] IBM Research – Zurich

marcel . armour . 2017 @ rhul . ac . uk
poe @ zurich . ibm . com

**Abstract.** This work introduces a new class of Algorithm Substitution Attack (ASA) on Symmetric Encryption Schemes. ASAs were introduced by Bellare, Paterson and Rogaway in light of revelations concerning mass surveillance. An ASA replaces an encryption scheme with a subverted version that aims to reveal information to an adversary engaged in mass surveillance, while remaining undetected by users. Previous work posited that a particular class of AEAD scheme (satisfying certain correctness and uniqueness properties) is resilient against subversion. Many if not all real-world constructions – such as GCM, CCM and OCB – are members of this class. Our results stand in opposition to those prior results. We present a potent ASA that generically applies to *any* AEAD scheme, is undetectable in all previous frameworks and which achieves successful exfiltration of user keys. We give even more efficient *non-generic* attacks against a selection of AEAD implementations that are most used in practice. In contrast to prior work, our new class of attack targets the decryption algorithm rather than encryption. We argue that this attack represents an attractive opportunity for a mass surveillance adversary. Our work serves to refine the ASA model and contributes to a series of papers that raises awareness and understanding about what is possible with ASAs.

**Keywords:** Algorithm Substitution Attacks · Privacy · Symmetric Encryption · Mass Surveillance

[⋆] 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). The research of Poettering was supported by the European Union's Horizon 2020 project FutureTPM (779391).

## 1 Introduction

The Snowden revelations in 2013 exposed that mass surveillance is a reality. They also showed that even sophisticated adversaries with large resources have been unable to break well established cryptographic primitives and hardness assumptions, shifting their focus to circumventing cryptography. Together, these two facts suggest that the study of subverted implementations of cryptographic primitives and protocols is a fruitful area of research; Rogaway has gone so far as to call it a moral imperative [22]. The reader is referred to the survey by Schneier et al. [27], which provides a broad overview of subversion of cryptography, with some useful case studies. 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* [29,30]. This area of study can be traced back to Simmons' work on *subliminal channels*, e.g. [28], undertaken in the context of nuclear non-proliferation during the Cold War. In the original conception, 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.

Preceding Work. Post-Snowden, work in this area was reignited by Bellare, Paterson and Rogaway (BPR) [6], 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) while remaining undetected by the user. There is a tension for 'Big Brother' between mounting a successful attack and being detected; clearly an attack that simply replaces the encryption algorithm with one that outputs the messages in plaintext would be devastating yet trivially detectable. BPR stipulate that subverted schemes should at the very least decrypt correctly (according to the unmodified specification) in order to have some measure of resistance to detection, going on to define the success probability of a mass surveillance adversary in carrying out a successful attack, as well as the advantage of a user in detecting that an attack is taking place. BPR [6] demonstrate an attack against randomized schemes that relies on influencing the randomness generated in the course of encryption. Their attack applies to a sub-class of randomized schemes satisfying a property they call 'coin-injectivity'. Lastly, BPR also establish a positive result that shows that under certain assumptions, it is possible for authenticated encryption schemes to provide resistance against subversion attacks.

Degabriele, Farshim and Poettering (DFP) [10] critiqued the definitions and underlying assumptions of BPR. Their main insight is that perfect decryptability —as mandated by BPR— is a very strong requirement and artificially limits the adversary's set of available strategies. In practice, a subversion with negligible failure probability should be considered effectively correct.[3] As DFP note, decryption failures may happen for reasons other than subverted encryption, and if they occur sporadically may easily go unnoticed. DFP demonstrate how this can be achieved with an input-triggered subversion, where the trigger is some input (message, associated data, nonce, or a combination thereof) that is difficult to guess, making detection practically impossible.

Bellare, Jaeger and Kane (BJK) [4] improved on the attack of BPR, giving an attack which is effective against all randomized schemes. Furthermore, whereas the attack of BPR is stateful and so vulnerable to detection through state reset, the BJK attack is stateless. BJK [4] furthermore formalised that the desired outcome of an ASA from the point of view of a mass surveillance adversary is successful key recovery.

In concurrent work, we study the effects of subverting the receiver in the setting of message authentication codes [1]. Using similar techniques, we provide ASAs that result in successful key exfiltration.

Contributions. Our work continues a line of investigation that serves to raise awareness of what is possible with ASAs, and highlights the importance of work countering subverted implementations. We consider ASAs from a new perspective that leads to results of practical importance. Recall that BPR established a covert channel through ciphertexts by manipulating the randomness generation; their model stipulated perfect decryptability, which resulted in their definitions being fragile. DFP identified

[3] This is analogous to the fundamental notion in cryptography that a symmetric encryption scheme be considered secure even in the presence of adversaries with negligible advantage.

this and proposed tolerating a (minimal) compromise of correctness, allowing trigger messages. We note that attacks employing trigger messages appear trivial to plant in formal security abstractions like IND-CPA where the adversary has full control over encrypted messages, associated data, and nonces. In practice, however, it is certainly questionable that adversaries have enough influence on any of the three to conduct DFP style attacks, as messages are chosen in special formats mandated by applications, nonces are implemented via counters, etc. We remove these dependencies, complementing the DFP approach, by attacking from a different angle: leaving perfect correctness intact, we (minimally) limit ciphertext integrity and establish a covert channel through decryption error events. Concretely, we manipulate the decryption algorithm to accept certain bogus ciphertexts. This requires the surveillance adversary to be able to observe whether a decryption implementation outputs a message or rejects the ciphertext. In many practical scenarios this is a mild assumption, for example if a decryption error results in a packet being dropped and automatically retransmitted. Furthermore, a subverted decryption algorithm could go beyond this by e.g. influencing timing information in future messages sent to the network. We conclude that this attack represents an attractive and easy to implement opportunity for a mass surveillance adversary.

Our results stand in opposition to previous work [6,10,4] which proposed subversion resilience of a large class of AEAD schemes to which many if not all real-world constructions such as GCM, CCM and OCB belong, as long as their nonces are generated deterministically via a shared state maintained by both encryptor and decryptor.[4] The key observation to resolve this apparent contradiction is that previous work has assumed, besides explicitly spelled out requirements like uniqueness of ciphertexts and perfect decryptability, implicit notions such as integrity of ciphertexts. In the ASA setting where undermining the confidentiality of an encryption scheme is the key goal of an adversary, it seems just as natural to assume that the adversary is also willing to compromise the integrity guarantees as well.

Related Work. We outlined the key publications on ASAs against symmetric encryption schemes above. Other works, briefly described here, consider subversion on different primitives and in differerent contexts. Berndt and Liskiewicz [7] reunite the fields of cryptography and steganography. Ateniese, Magri and Venturi [2] study ASAs on signature schemes. In a series of work, Russell, Tang, Yung and Zhou [23,24,25,26] consider ASAs on one-way functions, trapdoor one-way functions and key generation as well as defending randomized algorithms against ASAs. Goh, Boneh, Pinkas and Golle [17] show how to add key recovery to the SSL/TLS and SSH protocols. Dodis, Ganesh, Golovnev, Juels and Ristenpart [11] provide a formal treatment of backdooring PRGs, another form of subversion. Camenisch, Drijvers and Lehmann [9] consider Direct Anonymous Attestation in the presence of a subverted Trusted Platform Module. Cryptographic reverse firewalls [20,12,19] represent an architecture to counter ASAs via trusted code in network perimeter filters. Fischlin and Mazaheri show how to construct ASA-resistant encryption and signature algorithms given initial access to a trusted base scheme [16]. Fischlin, Janson and Mazaheri [15] show how to immunize (keyed and unkeyed) hash functions against subversion. Bellare, Kane and Rogaway [5] explore using large keys to prevent key exfiltration in the symmetric encryption setting. Bellare and Hoang [3] give public key encryption schemes that defend against the subversion of random number generators.

Structure. We first recall (Sect. 2) standard definitions for symmetric encryption schemes and their security. We next give definitions (Sect. 3) that provide a general framework in which to study ASAs. These have been refined and extended from prior work, crucially including the decryption oracle which had been ignored by previous work. Sect. 4 details our new type of attack, together with formal theorems quantifying the ability of an adversary to exfiltrate keys and the ability of the subversion to go undetected. We give two versions of our ASA: one for a passive adversary (the adversarial model considered by previous work), which we extend to a second ASA requiring an active trigger: a modified ciphertext provided to the decryption algorithm. We discuss the results of a proof-of-concept implementation in Sect. 5. Lastly, Sect. 6 explains how our attacks can be leveraged to compromise the security of popular practical schemes even more effectively, demonstrating how powerful ASAs become when conducted outside the clearly demarcated boundaries of a formal model. Concretely, we give evidence that ASAs against standardized AEAD constructions like GCM or OCB3 can be even more damaging than our attacks from Sect. 4.

[4] The members of this class are deterministic and satisfy certain technical correctness and uniqueness properties.

## 2 Notation and Definitions

Notation. For a natural number $k \in \mathbb{N}$, we let $[k] = \{0, 1, \ldots, k-1\}$. We refer to an element $x \in \{0,1\}^*$ as a string, and denote its length by $|x|$. By $\varepsilon$ we denote the empty string. The set of strings of length $\ell$ is denoted $\{0,1\}^\ell$. In addition we denote by $\perp \notin \{0,1\}^*$ a reserved special symbol. 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 $x = x[0] \ldots x[|x|-1]$. For two strings $x, x'$ we denote by $x \parallel x'$ their concatenation. If $S$ is a finite set, then $s \leftarrow_\$ S$ denotes choosing $s$ uniformly at random from $S$. If $\mathcal{A}$ is a randomized algorithm, we write $y \leftarrow_\$ \mathcal{A}(x)$ to indicate that it is invoked on input $x$ (and fresh random coins), and the result is assigned to variable $y$. In security games we write $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_c} \implies 1$ to denote the event that the adversary outputs 1 after being given access to the $c$ oracles.

PRFs. In Appendix A, we recall standard definitions for pseudorandom functions and permutations.

### 2.1 Symmetric Encryption

We focus on the most widespread and practically useful encryption primitive: Authenticated Encryption with Associated Data (AEAD). We recall standard definitions of (deterministic) nonce-based AEAD, as per [21].

AEAD. A symmetric encryption scheme $\Pi$ providing authenticated encryption with associated data is a triple of algorithms $(\Pi.\mathsf{Gen}, \Pi.\mathsf{Enc}, \Pi.\mathsf{Dec})$. Associated to $\Pi$ are two parameters, $\Pi.\mathsf{kl}$ and $\Pi.\mathsf{nl}$, representing the key length and the nonce length. The key generation algorithm $\Pi.\mathsf{Gen}$ is a probabilistic algorithm that takes as input the key length $\Pi.\mathsf{kl}$ and returns a key $k \in \{0,1\}^{\Pi.\mathsf{kl}}$. Often $\Pi.\mathsf{Gen}$ is taken as the algorithm choosing $k$ uniformly at random from $\{0,1\}^{\Pi.\mathsf{kl}}$. The encryption algorithm $\Pi.\mathsf{Enc}$ takes key $k$, message $m$, associated data $d$ and nonce $n \in \{0,1\}^{\Pi.\mathsf{nl}}$ to deterministically obtain ciphertext $c \leftarrow \Pi.\mathsf{Enc}(k, m, d; n)$. Decryption algorithm $\Pi.\mathsf{Dec}$ is deterministic and $\Pi.\mathsf{Dec}(k, c, d; n)$ returns either a message $m$ or the special symbol $\perp$. For simplicity, we assume that $|\Pi.\mathsf{Enc}(k, m, d; n)|$ is an affine function of the form $|m| + \tau$ where $\tau$ is some constant associated to the encryption scheme (all practical encryption schemes are of this type). We call $\tau$ the *stretch* of the encryption scheme. Lastly, where the context is clear, we drop the prefix $\Pi$.

**Definition 1.** *A symmetric encryption scheme $\Pi$ is said to be $\delta$-correct if for all tuples $(m, d; n)$ it holds that:*

$$\Pr\left[m \neq m' | c \leftarrow \mathsf{Enc}(k, m, d; n), m' \leftarrow \mathsf{Dec}(k, c, d; n)\right] \leq \delta,$$

*where the probability is taken over the set of keys $\{k : k \leftarrow_\$ \mathsf{Gen}(\mathsf{kl})\}$. If $\delta = 0$ the scheme is referred to as being perfectly correct.*

The classic privacy notion used for AEAD is indistinguishability from random bits under an adaptive chosen-plaintext-and-nonce attack, utilising standard game-based definitions. For the authenticity notion, we consider adversaries that aim to create (strong) forgeries. Security notions are as in [21]. Intuitively, the scheme provides confidentiality if the privacy advantage of any realistic adversary is negligible and authenticity if the forging advantage of any realistic adversary is negligible.

**Definition 2.** *The privacy advantage of an adversary $\mathcal{A}$ is given by*

$$\mathsf{Adv}^{\mathrm{priv}}_{\Pi}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathsf{Enc}(k,\cdot,\cdot;\cdot)} \implies 1 | k \leftarrow_\$ \mathsf{Gen}(\mathsf{kl})\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot;\cdot)} \implies 1\right],$$

*where the \$ oracle returns $c \leftarrow_\$ \{0,1\}^{|m|+\tau}$ for any query $\$(m, d; n)$. We assume that $\mathcal{A}$ is nonce-respecting; that is, $\mathcal{A}$ does not make two queries with the same nonce.*

**Definition 3.** *The authenticity advantage of an adversary $\mathcal{A}$ is given by*

$$\mathsf{Adv}^{\mathrm{auth}}_{\Pi}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathsf{Enc}(k,\cdot,\cdot;\cdot),\mathsf{Dec}(k,\cdot,\cdot;\cdot)} \textit{ forges} | k \leftarrow_\$ \mathsf{Gen}(\mathsf{kl})\right],$$

*where we say that $\mathcal{A}$ forges if it receives any $m' \neq \perp$ from $\mathsf{Dec}$ where we require that $(c, d; n)$ is not the result of an encryption query $(m, d; n)$. We assume that $\mathcal{A}$ is nonce-respecting; that is, $\mathcal{A}$ does not make two encryption queries with the same nonce.*

## 3 ASAs on Symmetric Encryption Schemes

We now outline the framework which will allow us to describe our concrete ASAs in Sect. 4. The aim of an ASA is to replace a given (symmetric encryption) scheme with a compromised version; if the original scheme is denoted $\Pi$, we write $\widetilde{\Pi}$ for its subversion. The attacker may choose to replace one component of the scheme, or multiple. We model the subverted scheme as having an embedded attacker key which is shared with an external (mass surveillance) adversary. This approach was first used by BPR [6]. From the attacker's perspective, the ASA should be undetectable by the user and result in effective surveillance. We formalise these notions as detectability and key recovery. Our definitions are inherited from prior work [6,4,10]. Whereas previous work assumed that only the encryption algorithm might be subverted, we have generalised the definitions to reflect the possibility that any component (one or multiple) of the symmetric encryption scheme could be subverted, and adapted to explicitly consider AEAD schemes. We broadly follow the notational choices of BJK [4].

ASA SYNTAX. An algorithm substitution attack $\mathsf{A}$ on a scheme $\Pi$ consists of a triple $(\mathsf{A.Gen}, \mathsf{A.Ext}, \widetilde{\Pi})$, where:

1. The attacker key generation algorithm $\mathsf{A.Gen}$ takes as input the security parameter $\mathsf{A.kl}$ and returns an attacker key $k_{\mathsf{A}} \in \{0,1\}^{\mathsf{A.kl}}$.
2. $\widetilde{\Pi} = (\widetilde{\Pi}.\mathsf{Gen}, \widetilde{\Pi}.\mathsf{Enc}, \widetilde{\Pi}.\mathsf{Dec})$ is a *subverted* symmetric encryption scheme.
   (a) The subverted key generation algorithm $\widetilde{\Pi}.\mathsf{Gen}$ is a probabilistic algorithm that takes as input the key length $\widetilde{\Pi}.\mathsf{kl}$ and the attacker key $k_{\mathsf{A}}$, returning a key $k \in \{0,1\}^{\widetilde{\Pi}.\mathsf{kl}}$.
   (b) The subverted encryption algorithm $\widetilde{\Pi}.\mathsf{Enc}$ takes the attacker key $k_{\mathsf{A}}$, user key $k$, message $m$, associated data $d$ and nonce $n \in \{0,1\}^{\widetilde{\Pi}.\mathsf{nl}}$, outputting ciphertext $c \leftarrow \widetilde{\Pi}.\mathsf{Enc}(k_{\mathsf{A}}, k, m, d; n)$.
   (c) The subverted decryption algorithm $\widetilde{\Pi}.\mathsf{Dec}(k_{\mathsf{A}}, k, c, d; n)$ returns either a message $m$ or the special symbol $\perp$.
3. The key extraction algorithm $\mathsf{A.Ext}$ takes as input $k_{\mathsf{A}}$ and has oracle access to both encryption and decryption oracles in the case of an active adversary, or to a transcript of ciphertexts in the case of a passive adversary. These notions are formalised in the key recovery game in Fig. 2. The output of this algorithm is a key $k \in \{0,1\}^{\widetilde{\Pi}.\mathsf{kl}}$.

We require that $\widetilde{\Pi}.\mathsf{kl} = \Pi.\mathsf{kl}$ and $\widetilde{\Pi}.\mathsf{nl} = \Pi.\mathsf{nl}$, as the subverted algorithm would otherwise be trivially detected. As in previous work, we assume throughout that the key generation is unsubverted, but we retain a syntax that allows for the more general case.

DETECTABILITY. In the formal notion of detectability, we allow a distinguisher $\mathcal{D}$ to interact with subverted encryption, subverted decryption and (for generality) subverted key generation. We assume that the distinguisher has access to its own reference copy of the unsubverted algorithms. It wins if it can distinguish between the base scheme and the subverted scheme in the game defined in Fig. 1. The detectability advantage of $\mathcal{D}$ with respect to $\Pi$, $\widetilde{\Pi}$ is given by

$$\mathsf{Adv}^{\mathrm{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) = 2 \cdot \Pr\left[\mathsf{Det}_{\Pi,\widetilde{\Pi}}(\mathcal{D})\right] - 1.$$

```
Game $\mathsf{Det}_{\Pi,\widetilde{\Pi}}(\mathcal{D})$
$k_{\mathsf{A}} \leftarrow_\$ \mathsf{A.Gen}(\mathsf{A.kl})$
$b \leftarrow_\$ \{0,1\}, b' \leftarrow \mathcal{D}^{\mathcal{O}_{\mathsf{Gen}},\mathcal{O}_{\mathsf{Enc}},\mathcal{O}_{\mathsf{Dec}}}$
return $(b = b')$

$\mathcal{O}_{\mathsf{Gen}}(\mathsf{kl})$
if $(b = 1)$ then $k \leftarrow_\$ \Pi.\mathsf{Gen}(\mathsf{kl})$
else $k \leftarrow_\$ \widetilde{\Pi}.\mathsf{Gen}(k_{\mathsf{A}}, \mathsf{kl})$
return $k$

$\mathcal{O}_{\mathsf{Enc}}(k, m, d; n)$
if $(b = 1)$ then $c \leftarrow \Pi.\mathsf{Enc}(k, m, d; n)$
else $c \leftarrow \widetilde{\Pi}.\mathsf{Enc}(k_{\mathsf{A}}, k, m, d; n)$
return $c$

$\mathcal{O}_{\mathsf{Dec}}(k, c, d; n)$
if $(b = 1)$ then $m \leftarrow \Pi.\mathsf{Dec}(k, c, d; n)$
else $m \leftarrow \widetilde{\Pi}.\mathsf{Dec}(k_{\mathsf{A}}, k, c, d; n)$
return $m$
```

**Fig. 1.** Game to define detectability advantage of $\mathcal{D}$ with respect to $\widetilde{\Pi}$, $\Pi$ and $q$.

This definition is adapted from strong undetectability of [4]. Notice that (informally) a ‘hard-to-detect’ subversion of a perfectly correct base scheme necessarily satisfies some correctness condition. To see this, suppose that the subversion does not satisfy $\delta$-correctness: it is detectable with probability at least $\delta$.

Key Recovery. Following [4], recovering the user’s secret key is a strong property for an attacker. We give two flavours of the key recovery game, one for passive adversaries $\mathsf{PassiveKR}$ and one for active adversaries $\mathsf{ActiveKR}$, as given in Fig. 2. In the passive case, we allow the adversary to observe ciphertexts and whether they are rejected. This is formalised through the transcript oracle $\mathcal{O}_{\mathsf{Trans}}$. For the active case, we allow the attacker to generate valid ciphertexts via $\mathcal{O}_{\mathsf{Enc}}$ and interact with a decryption oracle $\mathcal{O}_{\mathsf{Dec}}$ that reveals whether a submitted ciphertext is rejected. Both games are parametrised by a message sampler algorithm $\mathcal{M}$. Given its current state $\sigma$, $\mathcal{M}$ returns the next message with associated data $(m, d)$ to be encrypted, together with a nonce $n \in \{0,1\}^{\Pi.\mathsf{nl}}$ and an updated state. It represents the choice of messages made by the sender. For simplicity, we model $\mathcal{M}$ as non-adaptive and nonce-respecting. It could be argued that a more realistic model might take into account that the adversary could influence the user’s choice of messages to be encrypted. However, in constructing attacks we assume the weakest properties of the attacker.

$\mathsf{A}$ wins if $\mathsf{A.Ext}$ recovers the user’s key $k$ after interacting with the subverted encryption scheme. The key recovery advantage of $\mathsf{A}$ with respect to $\widetilde{\Pi}$ and $\mathcal{M}$ is given by

$$\mathsf{Adv}^{\mathrm{kr}}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A}) = \Pr\left[\mathsf{KR}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A})\right],$$

where $\mathsf{KR}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A})$ refers to the appropriate key recovery game according to whether the adversary is passive or active.

```
Game $\mathsf{ActiveKR}_{\widetilde{\Pi},\mathcal{M}}$
$k_\mathsf{A} \leftarrow \mathsf{A.Gen}(\mathsf{A.kl})$
$k \leftarrow \widetilde{\Pi}.\mathsf{Gen}(\widetilde{\Pi}.\mathsf{kl}), \sigma \leftarrow \varepsilon$
$k' \leftarrow \mathsf{A.Ext}^{\mathcal{O}_{\mathsf{Enc}},\mathcal{O}_{\mathsf{Dec}}}(k_\mathsf{A})$
return $(k' = k)$

$\mathcal{O}_{\mathsf{Enc}}()$
$(m, d, n, \sigma) \leftarrow_\$ \mathcal{M}(\sigma)$
$c \leftarrow \widetilde{\Pi}.\mathsf{Enc}(k_\mathsf{A}, k, m, d; n)$
return $(c, d; n)$

$\mathcal{O}_{\mathsf{Dec}}(c, d; n)$
$m \leftarrow \widetilde{\Pi}.\mathsf{Dec}(k_\mathsf{A}, k, c, d; n)$
return $(m = \perp)$
```

```
Game $\mathsf{PassiveKR}_{\widetilde{\Pi},\mathcal{M}}$
$k_\mathsf{A} \leftarrow \mathsf{A.Gen}(\mathsf{A.kl})$
$k \leftarrow \widetilde{\Pi}.\mathsf{Gen}(\widetilde{\Pi}.\mathsf{kl}), \sigma \leftarrow \varepsilon$
$k' \leftarrow \mathsf{A.Ext}^{\mathcal{O}_{\mathsf{Trans}}}(k_\mathsf{A})$,
return $(k' = k)$

$\mathcal{O}_{\mathsf{Trans}}()$
$(m, d, n, \sigma) \leftarrow_\$ \mathcal{M}(\sigma)$
$c \leftarrow \widetilde{\Pi}.\mathsf{Enc}(k_\mathsf{A}, k, m, d; n)$
$m \leftarrow \widetilde{\Pi}.\mathsf{Dec}(k_\mathsf{A}, k, c, d; n)$
return $(c, d, (m = \perp))$
```

**Fig. 2.** Game to define key recovery advantage of $\mathsf{A}$ with respect to $\widetilde{\Pi}$ and $\mathcal{M}$.

## 4 Mounting Attacks via Decryption Subversion

We now detail our ASAs, first for a passive surveillance adversary and then in the active case. It is easy to see that these are undetectable according to the models in the literature [6,4,10], as the encryption algorithm is not subverted.

In the passive attack, the decryption algorithm is subverted so that it rejects a fraction of valid ciphertexts, bounded by an attacker controlled parameter. In the active attack, it is subverted so that it accepts a (similarly bounded) fraction of invalid ciphertexts. In either case, this is done in such a way that an adversary, observing the rejected ciphertexts, can recover the user’s secret key. We note that both of our attacks are stateless, which not only allows for much easier back door implementation from a technical perspective but also should decrease the likelihood that an implemented attack is detected through code review or observing memory usage.

```
Algorithm $\widetilde{\Pi}.\mathsf{Dec}(k_{\mathsf{A}}, k, c, d; n)$
1: parse $F(k_{\mathsf{A}}, c \parallel d)$ as $i \parallel b$
2: if $(k[i] = b)$ then
3:     if $B(\delta)$ then
4:         return $\perp$
5: else
6:     return $\Pi.\mathsf{Dec}(k, c, d; n)$
```

```
Algorithm $\mathsf{A.Ext}^{\mathcal{O}_{\mathsf{Trans}}}(k_{\mathsf{A}})$
1: $\forall i \in [\Pi.\mathsf{kl}], \mathsf{key}[i] \leftarrow \star$
2: while $\exists j\colon \mathsf{key}[j] = \star$ do
3:     $(c, d, v) \leftarrow \mathcal{O}_{\mathsf{Trans}}$
4:     if $(v = 1)$ then
5:         parse $F(k_{\mathsf{A}}, c \parallel d)$ as $i \parallel b$
6:         if $(\mathsf{key}[i] = \star)$ then
7:             $\mathsf{key}[i] \leftarrow b$
8: return $\mathsf{key}$
```

**Fig. 3.** Passive Attack

### 4.1 Attack 1: Passive

Consider the following subversion of a given symmetric encryption scheme ($\Pi.\mathsf{Gen}$, $\Pi.\mathsf{Enc}, \Pi.\mathsf{Dec}$). Let $\widetilde{\Pi}.\mathsf{Gen} = \Pi.\mathsf{Gen}$ and $\widetilde{\Pi}.\mathsf{Enc} = \Pi.\mathsf{Enc}$. Let $\mathsf{A.Gen}$ choose a key $k_{\mathsf{A}}$ by $k_{\mathsf{A}} \leftarrow_{\$} \{0,1\}^{\mathsf{A.kl}}$. $\widetilde{\Pi}.\mathsf{Dec}$ takes the same input as $\Pi.\mathsf{Dec}$ together with the attacker key, and is defined below. The subverted decryptor utilises a pseudo-random function[5] $F$ with $F\colon \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^* \to [\Pi.\mathsf{kl}] \times \{0,1\}$. In $\mathsf{A.Ext}$, we use $\star$ as a ternary symbol (neither 0 nor 1) to keep track of which key bits have been collected. In line 3 of the algorithm for $\widetilde{\Pi}.\mathsf{Dec}$, we write $B(\delta)$ to denote a Bernoulli trial which returns 1 with probability $\delta$. $\mathsf{A.Ext}$ takes as input the attacker key and the transcript from the decryptor, consisting of pairs $(c, d, v)$ where $v$ is a bit representing whether or not the ciphertext decrypts to $\perp$.

**Theorem 1.** *Let $\Pi$ be a perfectly-correct symmetric encryption scheme and let $\ell = \Pi.\mathsf{kl}$. Let $\widetilde{\Pi}.\mathsf{Dec}$ and $\mathsf{A.Ext}$ be defined as in Fig. 3. Let $\mathcal{M}$ be a message sampling algorithm, and $F\colon \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^* \to [\ell] \times \{0,1\}$ be a PRF with $\mathsf{Adv}^{\mathrm{prf}}_{F}(\mathcal{F}) < \epsilon$ for all PPT adversaries $\mathcal{F}$. Then*

1. *$\mathsf{Adv}^{\mathrm{kr}}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A}) \geq 1 - \ell e^{-\frac{q\delta}{2\ell}}$, where $q$ is the number of queries that $\mathsf{A.Ext}$ makes to the transcript oracle.*
2. *For all distinguishers $\mathcal{D}$, $\mathsf{Adv}^{\mathrm{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) \leq \frac{\delta q}{2}(1 + \epsilon)$ where $\mathcal{D}$ makes $q$ queries to its decryption oracle.*

*Proof (1).* We use a combinatorial argument. Notice that this is essentially a coupon collection problem. We are looking for the probability that every key bit has been exfiltrated. If we fix $i$ key bits that are not exfiltrated, there are $\binom{\ell}{i}$ ways to choose those fixed key bits. The probability that (at least) $i$ of the key bits have not been exfiltrated is given by $\binom{\ell}{i}(1 - \frac{i\delta}{2\ell})^q$. Using the principle of inclusion exclusion, the probability that no key bit has not been exfiltrated is given by

$$\begin{aligned}\mathsf{Adv}^{\mathrm{kr}}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A}) &= \sum_{i=0}^{\ell} (-1)^i \binom{\ell}{i} \left(1 - \frac{i\delta}{2\ell}\right)^q \\ &\geq 1 - \ell \left(1 - \frac{\delta}{2\ell}\right)^q \\ &\geq 1 - \ell e^{-\frac{q\delta}{2\ell}}\end{aligned}$$

*(2).* Clearly, the only way to distinguish between $\Pi$ and $\widetilde{\Pi}$ is to observe $\widetilde{\Pi}.\mathsf{Dec}$ output $\perp$. Thus in order to distinguish, $\mathcal{D}$ must find $(m, d; n)$ such that $\perp = \mathcal{O}_{\mathsf{Dec}}(k, c, d; n)$ for $c \leftarrow \Pi.\mathsf{Enc}(k, m, d; n)$. This reduces to $\mathcal{D}$ finding some $c \parallel d$ such that $F(k_{\mathsf{A}}, c \parallel d) = i \parallel k[i]$ for some index $i$. Call this event $W$. Notice that for any $F$ it holds that for all $k_{\mathsf{A}}, c, d$ we have $F(k_{\mathsf{A}}, c \parallel d) = i \parallel b$ for some index $i$ and bit $b$.

[5] See Appendix A for definitions of pseudo-random functions and permutations.

We note that $\Pr[W] \leq \Pr[\mathsf{PRF}_F(\mathcal{F})]$ for all PRF adversaries $\mathcal{F}$. If not, it would be possible for $\mathcal{F}$ to act as a challenger to $\mathcal{D}$ and win its prf game whenever $W$ occurs. Thus,

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) &= \Pr\left[\mathsf{Det}_{\Pi,\widetilde{\Pi}}(\mathcal{D})|b=1\right] + \Pr\left[\mathsf{Det}_{\Pi,\widetilde{\Pi}}(\mathcal{D})|b=0\right] - 1 \\
&= 1 - (1 - \delta \cdot \Pr[W])^q \\
&\leq 1 - (1 - \delta \cdot \Pr[\mathsf{PRF}_F(\mathcal{D})])^q \\
&\leq 1 - \left(1 - \frac{\delta}{2}(1 + \mathsf{Adv}^{\mathrm{prf}}_F(\mathcal{D}))\right)^q \\
&\leq 1 - \left(1 - \frac{\delta}{2}(1+\epsilon)\right)^q \\
&\leq \frac{\delta q}{2}(1+\epsilon).
\end{aligned}$$

□

Remark. Whereas detectability does depend on the security of the PRF, the PRF can be extremely weak without impacting the adversary's key recovery advantage. If the base scheme Π's ciphertexts are indistinguishable from random, then the PRF could simply choose the first $\lceil \log(\ell) \rceil + 1$ many bits of the ciphertext. This seems paradoxical, as strong privacy security is usually a desirable property but here allows the ASA to be successful.

We note that in practice, the subverted decryption algorithm $\widetilde{\Pi}.\mathsf{Dec}$ can be made more effective in a number of ways, for example by exfiltrating more than one bit at a time. Furthermore, the model is very conservative and in practice it may be possible for $\mathsf{A.Ext}$ to observe a number of error messages following [8].

## 4.2 Attack 2: Active

We show how devastating an attack can be if we allow the adversary to have only slightly more capabilities. Imagine that Alice communicates with Bob. A passive adversary can observe ciphertexts from Alice to Bob. In addition, an active adversary can replace ciphertexts in transmission and submit its own (forged) ciphertexts to Bob. We present an algorithm substitution attack that targets Bob's decryption algorithm, and which requires the adversary to send Bob bogus ciphertexts (derived from genuine ciphertexts) that reveal Bob's secret key using decryption errors. Normally, these bogus ciphertexts would be rejected as they are unlikely to decrypt correctly. However, if the decryptor is subverted then the bogus ciphertext can be accepted, creating a covert channel that will allow the key to be exfiltrated. From the point of view of a mass surveillance adversary this is an attractive prospect: having passively collected all communications, triggered by some suspicion they can now target Alice and Bob's communication. By recovering Bob's key they may now decrypt all of the stored communication between Alice and Bob (and indeed from Bob to Alice as well).

```
Algorithm $\widetilde{\Pi}.\mathsf{Dec}(k_\mathsf{A}, k, c, d; n)$
 1: $m \leftarrow \Pi.\mathsf{Dec}(k, c, d; n)$
 2: if $(m \neq \perp)$ then
 3:     return $m$
 4: $\widetilde{c} \leftarrow E^{-1}(c)$
 5: $m \leftarrow \Pi.\mathsf{Dec}(k, \widetilde{c}, d; n)$
 6: if $(m = \perp)$ then
 7:     return $\perp$
 8: parse $F(k_\mathsf{A}, \widetilde{c} \parallel d)$ as $i \parallel b$
 9: if $(k[i] = b)$ then
10:     return $m$
11: else
12:     return $\perp$
```

```
Algorithm $\mathsf{A.Ext}^{\mathcal{O}_\mathsf{Enc}, \mathcal{O}_\mathsf{Dec}}(k_\mathsf{A})$
 1: $\forall i \in [\Pi.\mathsf{kl}], \mathsf{key}[i] \leftarrow \star$
 2: while $\exists j\colon \mathsf{key}[j] = \star$ do
 3:     $(c, d; n) \leftarrow \mathcal{O}_\mathsf{Enc}()$
 4:     parse $F(k_\mathsf{A}, c \parallel d)$ as $i \parallel b$
 5:     if $(\mathsf{key}[i] = \star)$ then
 6:         $\widetilde{c} \leftarrow E(c)$
 7:         $v \leftarrow \mathcal{O}_\mathsf{Dec}(\widetilde{c}, d; n)$
 8:         $\mathsf{key}[i] \leftarrow b \oplus v$
 9: return key
```

**Fig. 4.** Active Attack

The formal model is that described in the game $\mathsf{ActiveKR}_{\widetilde{\Pi},\mathcal{M}}$ in Fig. 2. The adversary $\mathsf{A.Ext}$ crafts special messages using a pseudo-random permutation $E$ under the attacker key. We let $E : \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^{|m|+\tau} \to \{0,1\}^{|m|+\tau}$. In order to obtain ciphertexts whose lengths match the input of $E$, we require $\mathcal{M}$ to output messages of a fixed size $|m|$. This is a convenient simplification: in practice, for longer ciphertexts the decryptor can simply use the first $|m| + \tau$ bits as input to $E$; shorter ciphertexts can be decrypted correctly. The security of $E$ will determine how easily the distinguisher $\mathcal{D}$ will be able to recreate a special message to trigger $\widetilde{\Pi}$. From the point of view of the subverted decryption algorithm $\widetilde{\Pi}.\mathsf{Dec}$, the PRP $E$ could practically be instantiated using $\Pi$. Furthermore, $\widetilde{\Pi}.\mathsf{Dec}$ makes use of a PRF $F$ to determine whether or not to reject submitted ciphertexts. We let $F\colon \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^* \to [\Pi.\mathsf{kl}] \times \{0,1\}$.

**Theorem 2.** *Let $\Pi$ be a perfectly-correct symmetric encryption scheme and let $\ell = \Pi.\mathsf{kl}$. Let $\widetilde{\Pi}.\mathsf{Dec}$ and $\mathsf{A.Ext}$ be defined as in Fig. 4. Let $\ell = \Pi.\mathsf{kl}$ and $\mathsf{Adv}^{\mathrm{auth}}_{\Pi} < \epsilon$. Let $F\colon \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^* \to [\ell] \times \{0,1\}$ be a PRF, and let $E$ be a PRP with $E : \{0,1\}^{\mathsf{A.kl}} \times \{0,1\}^{|m|+\tau} \to \{0,1\}^{|m|+\tau}$ and $\mathsf{Adv}^{\mathrm{prp}}_{\mathcal{D}}(E) < \epsilon'$. Finally, let $\mathcal{M}$ a message sampler outputting messages of fixed length $|m|$. Then*

1. *$\mathsf{Adv}^{\mathrm{kr}}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A}) \geq 1 - \ell e^{-\frac{q}{\ell}(1-\epsilon)}$, where $\mathsf{A.Ext}$ makes exactly $\Pi.\mathsf{kl}$ calls to the decryption oracle and $q$ calls to the encryption oracle.*
2. *For every Distinguisher $\mathcal{D}$, $\mathsf{Adv}^{\mathrm{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) \leq \frac{q}{2^\tau} + \epsilon'$, where $\mathcal{D}$ makes $q$ queries to its decryption oracle.*

*Proof (1).* We use the same combinatorial argument again. This time, the probability that (at least) $i$ of the key bits have not been correctly exfiltrated is given by $\binom{\ell}{i}\left[(1-\frac{i}{\ell}) + \frac{\alpha i}{2\ell}\right]^q$. Here $\alpha$ is the probability that $\Pi.\mathsf{Dec}(k, \widetilde{c}, d; n) \neq \perp$ given that $E^{-1}(\widetilde{c}) = j \parallel k[j]$ for $j$ in the set of indices being counted. We note that $\mathsf{Adv}^{\mathrm{auth}}_{\Pi} \geq \alpha$.

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{kr}}_{\widetilde{\Pi},\mathcal{M}}(\mathsf{A}) &= \sum_{i=0}^{\ell}(-1)^i\binom{\ell}{i}\left[(1-\frac{i}{\ell}) + \frac{\alpha i}{2\ell}\right]^q \\
&\geq 1 - \ell\left(1 + \frac{1}{\ell}(\frac{\alpha}{2} - 1)\right)^q \\
&\geq 1 - \ell e^{-\frac{q}{\ell}(1-\frac{\alpha}{2})} \\
&\geq 1 - \ell e^{-\frac{q}{\ell}(1-\epsilon)}.
\end{aligned}$$

*(2).* Once again, the only way to distinguish between $\Pi$ and $\widetilde{\Pi}$ is by observing $\widetilde{\Pi}.\mathsf{Dec}$ accepting a forged ciphertext. To do this, the distinguisher $\mathcal{D}$ must find some ciphertext $c$ with associated data $d$ such that $F(k_{\mathsf{A}}, \widetilde{c} \parallel d) = i \parallel k[i]$ for some $i \in [\ell]$ and where $\widetilde{c} = E^{-1}(c)$. Thus

$$\Pr\left[\mathsf{Det}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) | b = 0\right] \leq \Pr\left[\begin{array}{l}\mathcal{D} \text{ finds } c \text{ with } E^{-1}(c) = \widetilde{c} \text{ for some } \widetilde{c} \\ \quad \text{with } \Pi.\mathsf{Dec}(k, \widetilde{c}, d; n) \neq \perp, \text{ for some } d, n\end{array}\right]$$

Consider the following game, which we will refer to as the pre-image game. For $b \in \{0,1\}$ we define experiment $b$ as follows:
(a) The challenger selects $f \in \mathrm{Perms}_{|m|+\tau}$ in the following way:
  – if (b = 0) then $k \leftarrow_\$ \{0,1\}^{\mathsf{A.kl}}$, $f \leftarrow E(k, \cdot)$;
  – if $(b = 1)$ then $f \leftarrow_\$ \mathrm{Perms}_{|m|+\tau}$.
(b) The adversary $\mathcal{D}$ submits a sequence of queries $c_1, c_2, \ldots, c_q$ to the challenger and receives $c'_i = f^{-1}(c_i)$ for $i \in [q]$.

For $b \in \{0,1\}$, let $W_b$ be the event that $\mathcal{D}$ outputs 1 in experiment $b$; $\mathcal{D}$ outputs 1 if for some $d, n$, $\Pi.\mathsf{Dec}(k, c'_i, d; n) \neq \perp$. The advantage of $\mathcal{D}$ in the pre-image game is clearly less than its advantage in distinguishing a PRP from a random permutation. To see this, given $\mathcal{D}$ with some advantage playing the pre-image game we can construct an adversary $\mathcal{B}$ acting as a challenger to $\mathcal{D}$ such that $\mathcal{B}$ outputs 1 in the distinguishing game $\mathsf{PRP}_E(\mathcal{B})$ whenever $\mathcal{D}$ does in the pre-image game. Thus,

$$\Pr[W_0] - \Pr[W_1] \leq \mathsf{Adv}^{\mathrm{prp}}_{\mathcal{D}}(E).$$

Noting that $\Pr[W_1] = \frac{q}{2^\tau}$, we conclude that

$$\mathsf{Adv}^{\mathrm{det}}_{\Pi,\widetilde{\Pi}}(\mathcal{D}) \leq \Pr[W_0] \leq \Pr[W_1] + \mathsf{Adv}^{\mathrm{prp}}_{\mathcal{D}}(E) \leq \frac{q}{2^\tau} + \epsilon' \qquad \square$$

## 5 Implementation

We implemented our attacks in proof-of-concept code (written in Python 3) to verify their functionality and effectiveness.[6] The particular AEAD scheme we attack is AES-GCM [14], using black-box access to the implementation provided by [31]. We simulated both active and passive attacks 10,000 times, and recorded the number of queries for successful extraction of a 128-bit key (thus, $\ell = 128$). Messages, nonces and associated data were generated using the function `random.getrandbits` from the `Crypto.Random` library. Appendix B shows the distribution (in blue) of the recorded number of queries $q$, and (in red) the cumulative success probability as a function of $q$. The plots are reproduced below in Fig. 5. Our results confirm the theoretical estimates from Thm. 1 and 2; in particular, the exponential success rate. While the attacks have different application and success profiles, both reliably recover keys.

Passive. The expected number of calls to the transcript oracle for successful exfiltration is given by $\frac{2\ell}{\delta}\sum_{i=1}^{\ell}\frac{1}{i}$ (see proof of Thm. 1). We set $\delta = 0.1$ for illustration. This gives us an expected value of $q = 13910$ compared to the recorded mean of 13920.59. Alternatively, the result from Thm. 1 gives a key recovery advantage of $\approx 1/2$ with $q = 14000$, compared to the recorded median of 13380 (the discrepancy is due to the exponential approximation in the proof).

Active. We assume that for AES-GCM, $\mathsf{Adv}_{\Pi}^{\text{auth}} \approx 0$ and set $\epsilon = 0$. The expected number of encryption calls for successful exfiltration is then $\ell\sum_{i=1}^{\ell}\frac{1}{i}$ (see proof of Thm. 2). This gives an expected value of $q = 696$ compared to the recorded mean of 695.05. Alternatively, the result from Thm. 2 gives a key recovery advantage of $\approx 1/2$ with $q = 710$ compared to the recorded median of 670 (again, the difference is due to exponential approximation).

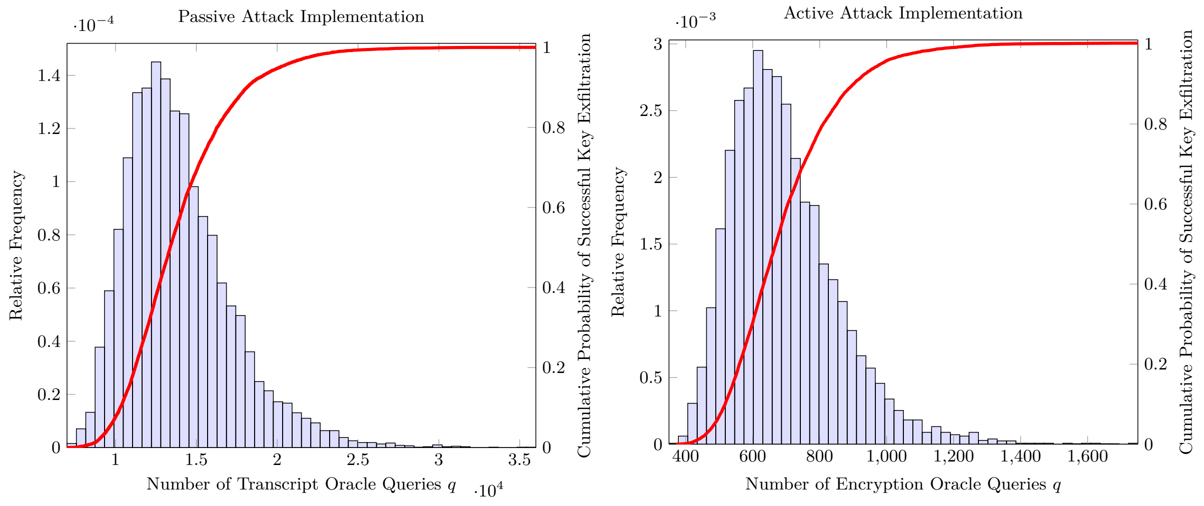

**Fig. 5.** Results of running an implementation of the active attack 10,000 times. See Appendix B for full-size versions. **Left:** Passive, see Fig. 7. **Right:** Active, see Fig. 8.

## 6 Breaking Security without Extracting the Full Key

The attacks presented in Sect. 4 are generic in that they are independent of the targeted AEAD scheme. Our approach, in common with previous work, was to extract the full key with which the AEAD instance is operated. Message recovery follows immediately by the definition of correctness. From this it is tempting to conclude that choosing longer keys, e.g. 256 bits instead of 128 in the case of AES-based encryption, gives better security against ASAs. In this section we show that this intuition is not necessarily correct. As we detail, many current AEAD schemes have inner building blocks that maintain their own secret

[6] We are happy share our source code. Please contact the authors.

values, and scaling up key sizes does not automatically also increase the sizes of these internal values. Note that ASAs in the style proposed in the previous section could easily be adapted to leak this internal information instead of the key. As the recovery of such values might not always directly lead to full message recovery, the assessment of whether the resulting overall attack is more or less effective than our generic attacks has to be made on a per scheme basis. We exemplify this on the basis of two of the currently best-performing AES-based AEAD schemes: GCM [13] and OCB3 [18]. In both cases, the size of the crucial internal value and the block size of the cipher have to coincide and the latter value is fixed to 128 bits for AES (independently of key size).

AES-GCM. We consider the following abstraction of GCM. The AEAD key $k$ is used directly to create an instance $E$ of the AES blockcipher. To encrypt a message $m$ with respect to associated data $d$ and nonce $n$, $E$ is operated in counter mode, giving a pad $E(n+1) \parallel E(n+2) \parallel \ldots$, where a specific nonce encoding ensures there are no collisions between counter values of different encryption operations. The first part $c_1$ of the ciphertext $c = c_1c_2$ is obtained by XORing the pad into the message, and finally the authentication tag $c_2$ is derived by computing $c_2 \leftarrow E(n)+H_h(d, c_1)$. Here $H_h$ is an instance of a universal hash function $H$ indexed (that is, keyed) with the 128-bit value $h = E(0^{128})$. Concretely, $H_h(d, c_1) = \sum_{i=1}^{l} v_i h^{l-i+1}$, where coefficients $v_1, \ldots, v_l$ are such that a prefix $v_1 \ldots v_j$ is a length-padded copy of the associated data $d$, the middle part $v_{j+1} \ldots v_{l-1}$ is a length-padded copy of ciphertext component $c_1$, and the last item $v_l$ is an encoding of the lengths of $d$ and $c_1$. The addition and multiplication operations deployed in this computation are those of a specific representation of the Galois field $\mathrm{GF}(2^{128})$.

In executing a practical algorithm substitution attack against AES-GCM, it might suffice to leak the value $h$ (which has length 128 independently of the AES key length, and furthermore stays invariant across encryption operations). The insight is that if the key of a universal hash function is known, then it becomes trivial to compute collisions. Concretely, assume the adversary is provided with the AES-GCM encryption $c = c_1c_2 = \mathsf{Enc}(k, m, d; n)$ for unknown $k, m$ but chosen $d, n$. Then by the above we have $c_2 = R + \sum_{i=1}^{j} v_i h^{l-i+1}$ where the coefficients $v_1 \ldots v_j$ are an encoding of $d$ and $R$ is some residue. If, having been successfully leaked by the ASA, the internal value $h$ is known, by solving a linear equation it is easy to find an associated data string $d' \neq d$, $|d'| = |d|$, such that for its encoding $v'_1 \ldots v'_j$ we have $\sum_{i=1}^{j} v'_i h^{l-i+1} = \sum_{i=1}^{j} v_i h^{l-i+1}$. Overall this means that we have found $d' \neq d$ such that $\mathsf{Enc}(k, m, d'; n) = c = \mathsf{Enc}(k, m, d; n)$. In a CCA attack the adversary can thus query for the decryption of $c$ with associated data $d'$ and nonce $n$, and thus fully recover the target message $m$. We finally note that this attack can be directly generalized to one where also the $c_1$ and $c_2$ components are modified, resulting in the decryption of a message $m' \neq m$ for which the XOR difference between $m = m'$ is controlled by the adversary.

OCB3. Multiple quite different versions of the OCB encryption scheme exist, but a common property is that the associated data input is incorporated via 'ciphertext translation' [21]. To encrypt a message $m$ under key $k$ with associated data $d$ and nonce $n$, in a first step the message $m$ is encrypted with a pure AE scheme (no AD!) to an intermediate ciphertext $c^* \leftarrow \mathsf{Enc}^*(k, m; n)$. Then to obtain the final ciphertext $c$, a pseudo-random function value $F_k(d)$ of the associated data string is XORed into the trailing bits of $c^*$. Concretely, in OCB3 we have $F_k(d) = \sum_{i=1}^{l} E(v_i + C_i)$ where all addition operations are XOR combinations of 128 bit values, $E(\cdot)$ stands for AES enciphering with key $k$, values $v_1, \ldots, v_l$ represent a length-padded copy of associated data $d$, and coefficients $C_1, \ldots, C_l$ are (secret) constants deterministically derived from the value $L = E(0^{128})$.

In the context of an ASA we argue that it is sufficient to leak the 128 bit value $L$. The attack procedure is, roughly, as in the AES-GCM case. Assume the adversary is provided with the OCB3 encryption $c = \mathsf{Enc}(k, m, d; n)$ for unknown $k, m$ but chosen $d, n$, and assume the adversary knows $L$ and thus $C_1, \ldots, C_l$. Now let $1 \leq s < t \leq l$ be any two indices, let $\Delta = C_s + C_t$ and let $d' \neq d$, $|d'| = |d|$, be the associated data string with encoding $v'_1, \ldots, v'_l$ such that we have $v'_s = v_t + \Delta$ and $v'_t = v_s + \Delta$ and $v'_i = v_i$ for all $i \neq s, t$. Then we have $E(v'_s + C_s) = E(v_t + \Delta + C_s) = E(v_t + C_t)$ and $E(v'_t + C_t) = E(v_s + \Delta + C_t) = E(v_s + C_s)$, which leads to $F_k(d) = F_k(d')$ and ultimately $\mathsf{Enc}(k, m, d'; n) = \mathsf{Enc}(k, m, d; n)$. In a CCA attack environment, this can immediately be leveraged to the full recovery of $m$. As in the AES-GCM case, we note that many variants of our attack exist (against all versions of OCB), including some that manipulate message bits in a controlled way.

## 7 Conclusion

Previous models of Algorithm Substitution Attack were founded on a number of implicit assumptions: notions of integrity and authenticated encryption, for example, were implicit in the work of BPR but remained just outside the boundary of their model. This work challenges the assumption of previous work that the decryptor can only be subverted in tandem with the encryptor. Targeting decryption only, we provide two examples of a new class of ASA that provides a mass surveillance adversary with a powerful and attractive strategy to compromise the confidentiality of mass communication.

## References

1. Armour, M., Poettering, B.: Substitution attacks against message authentication. IACR Trans. Symmetric Cryptol. ?(?), ?–? (2019), (to appear)
2. Ateniese, G., Magri, B., Venturi, D.: Subversion-resilient signature schemes. In: Ray, I., Li, N., Kruegel, C. (eds.) ACM CCS 2015: 22nd Conference on Computer and Communications Security. pp. 364–375. ACM Press (Oct 2015)
3. 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)
4. 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)
5. Bellare, M., Kane, D., Rogaway, P.: Big-key symmetric encryption: Resisting key exfiltration. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 373–402. Springer, Heidelberg (Aug 2016)
6. 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)
7. 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)
8. Boldyreva, A., Degabriele, J.P., Paterson, K.G., Stam, M.: On symmetric encryption with distinguishable decryption failures. In: Moriai, S. (ed.) Fast Software Encryption – FSE 2013. Lecture Notes in Computer Science, vol. 8424, pp. 367–390. Springer, Heidelberg (Mar 2014)
9. Camenisch, J., Drijvers, M., Lehmann, A.: Anonymous attestation with subverted TPMs. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017, Part III. Lecture Notes in Computer Science, vol. 10403, pp. 427–461. Springer, Heidelberg (Aug 2017)
10. 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)
11. Dodis, Y., Ganesh, C., Golovnev, A., Juels, A., Ristenpart, T.: A formal treatment of backdoored pseudorandom generators. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part I. Lecture Notes in Computer Science, vol. 9056, pp. 101–126. Springer, Heidelberg (Apr 2015)
12. 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)
13. Dworkin, M.: NIST special publication 800-38: Recommendation for block cipher modes of operation. US National Institute of Standards and Technology (2001), https://dx.doi.org/10.6028/NIST.SP.800-38D
14. Dworkin, M.J.: SP 800-38D: Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. Tech. rep., National Institute of Standards & Technology, Gaithersburg, MD, United States (2007), http://dx.doi.org/10.6028/NIST.SP.800-38D
15. Fischlin, M., Janson, C., Mazaheri, S.: Backdoored hash functions: immunizing HMAC and HKDF. In: 2018 IEEE 31st Computer Security Foundations Symposium (CSF). pp. 105–118. IEEE (2018)
16. Fischlin, M., Mazaheri, S.: Self-guarding cryptographic protocols against algorithm substitution attacks. In: 2018 IEEE 31st Computer Security Foundations Symposium (CSF). pp. 76–90. IEEE (2018)
17. Goh, E.J., Boneh, D., Pinkas, B., Golle, P.: The design and implementation of protocol-based hidden key recovery. In: Boyd, C., Mao, W. (eds.) ISC 2003: 6th International Conference on Information Security. Lecture Notes in Computer Science, vol. 2851, pp. 165–179. Springer, Heidelberg (Oct 2003)
18. Krovetz, T., Rogaway, P.: The OCB authenticated-encryption algorithm (2014), https://tools.ietf.org/html/rfc7253

19. 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)
20. 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)
21. Rogaway, P.: Nonce-based symmetric encryption. In: Roy, B.K., Meier, W. (eds.) Fast Software Encryption – FSE 2004. Lecture Notes in Computer Science, vol. 3017, pp. 348–359. Springer, Heidelberg (Feb 2004)
22. Rogaway, P.: The moral character of cryptographic work. Cryptology ePrint Archive, Report 2015/1162 (2015), http://eprint.iacr.org/2015/1162
23. 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)
24. 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), http://eprint.iacr.org/2016/530
25. 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)
26. 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)
27. Schneier, B., Fredrikson, M., Kohno, T., Ristenpart, T.: Surreptitiously weakening cryptographic systems. Cryptology ePrint Archive, Report 2015/097 (2015), http://eprint.iacr.org/2015/097
28. 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)
29. 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)
30. 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)
31. Zhu, B.: AES-GCM-Python (2013), https://github.com/bozhu/AES-GCM-Python/blob/master/aes_gcm.py

## A Pseudo-random Functions and Permutations

We recall standard notions of pseudo-random functions and permutations.

```
Game $\mathsf{PRF}_F(\mathcal{F})$
$L \leftarrow_\$ \{0,1\}^\ell, S \leftarrow \emptyset$
$b \leftarrow_\$ \{0,1\}, b' \leftarrow \mathcal{F}^{\mathcal{O}_{\mathsf{Fun}}}$
return $(b = b')$

$\mathcal{O}_{\mathsf{Fun}}(s)$
if $(b = 1)$ then $y_s \leftarrow F(L, s)$
else
    if $s \notin S$ then $y_s \leftarrow_\$ R$
    $S \leftarrow S \cup \{s\}$
return $y_s$

Game $\mathsf{PRP}_E(\mathcal{F})$
$L \leftarrow_\$ \{0,1\}^\ell, f \leftarrow \mathrm{Perms}_n$
$b \leftarrow_\$ \{0,1\}, b' \leftarrow \mathcal{F}^{\mathcal{O}_{\mathsf{Perm}}}$
return $(b = b')$

$\mathcal{O}_{\mathsf{Perm}}(s)$
if $(b = 1)$ then $y_s \leftarrow E(L, s)$
else
    $y_s \leftarrow f(s)$
return $y_s$
```

**Fig. 6.** Game to define prf and prp advantage of $\mathcal{F}$ with respect to $F$.

**Definition 4.** *A keyed pseudo-random function (PRF) for range $R$ is an efficiently computable function $F\colon \{0,1\}^\ell \times \{0,1\}^* \to R$ taking a key $L \in \{0,1\}^\ell$ and input $s \in \{0,1\}^*$ to return an output $F(L,s) \in R$. Consider game $\mathsf{PRF}_F(\mathcal{F})$ in Fig. 6 associated to $F$ and adversary $\mathcal{F}$. Let*

$$\mathsf{Adv}_F^{\mathrm{prf}}(\mathcal{F}) = 2 \cdot \Pr\left[\mathsf{PRF}_F(\mathcal{F})\right] - 1$$

*be the prf advantage of adversary $\mathcal{F}$ against function $F$. Intuitively, the function is pseudo-random if the prf advantage of any realistic adversary is negligible.*

**Definition 5.** *A keyed pseudo-random permutation (PRP) is an efficiently computable function $E\colon \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ taking a key $L \in \{0,1\}^\ell$ and input $s \in \{0,1\}^n$ to return an output $F(L,s) \in \{0,1\}^n$. We require that any keyed instance of $E$ is a permutation and also that its inverse $E^{-1}$ is efficiently computable. Consider game $\mathsf{PRP}_E(\mathcal{F})$ in Fig. 6 associated to $E$ and adversary $\mathcal{F}$, where $\mathrm{Perms}_n$ is the set of permutations on $\{0,1\}^n$. Let*

$$\mathsf{Adv}_E^{\mathrm{prp}}(\mathcal{F}) = 2 \cdot \Pr\left[\mathsf{PRP}_E(\mathcal{F})\right] - 1$$

*be the prp advantage of adversary $\mathcal{F}$ against function $E$. Intuitively, the permutation is pseudo-random if the prp advantage of any realistic adversary is negligible.*

## B Results

This section gives full size versions of the plotted results from Sect. 5.

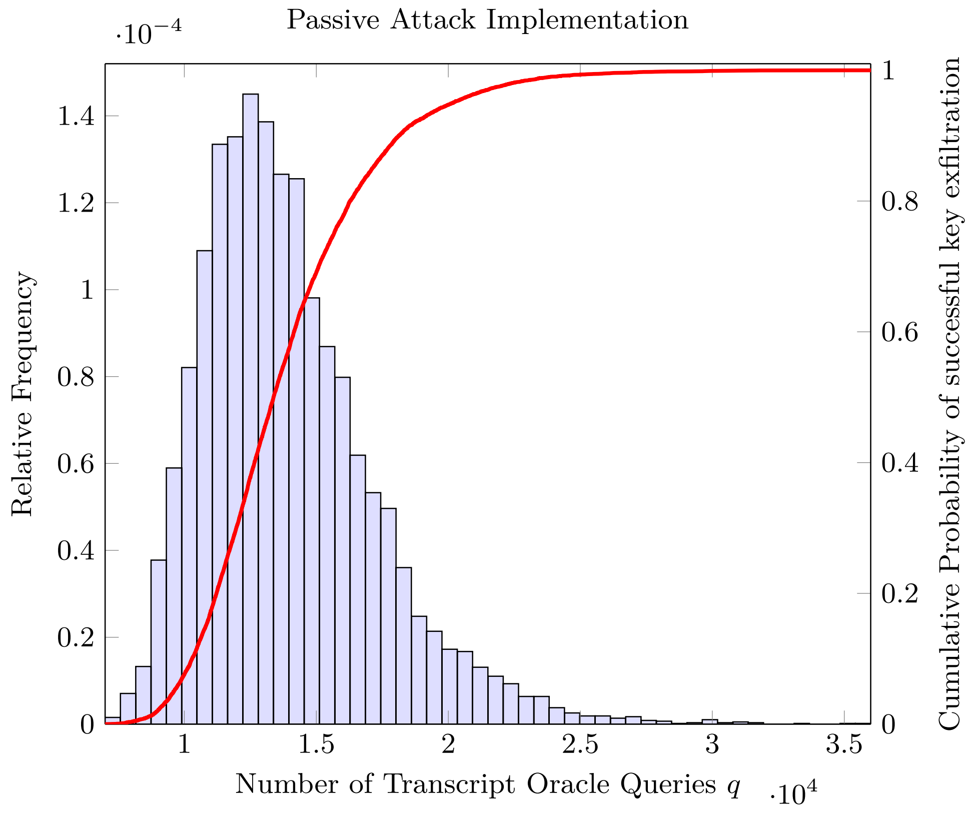

**Fig. 7.** Results of running an implementation of the passive attack 10,000 times. Key length $\ell = 128$, and parameter $\delta = 0.1$. **Left axis:** The blue histogram shows the distribution of the number of queries required for successful key exfiltration. The data has been sorted into 50 bins. **Right axis:** The red curve shows the cumulative probability of successful key exfiltration against $q$.

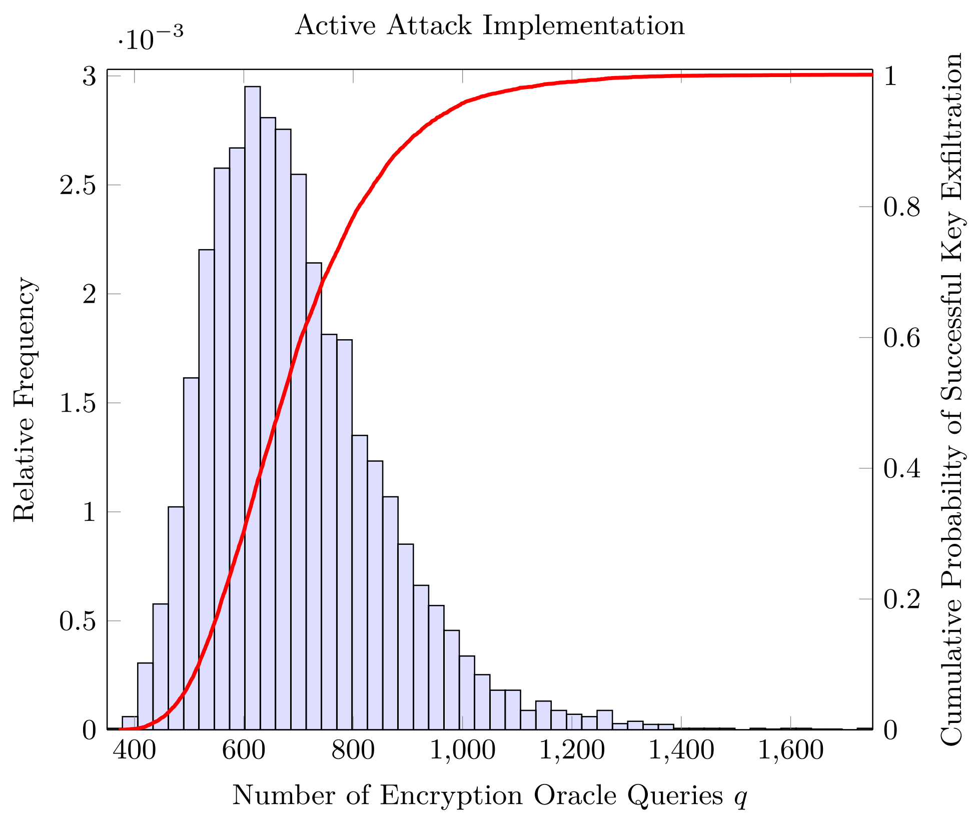

**Fig. 8.** Results of running an implementation of the active attack 10,000 times with key length $\ell = 128$. **Left axis:** The blue histogram shows the distribution of the number of queries required for successful key exfiltration. The data has been sorted into 50 bins. **Right axis:** The red curve shows the cumulative probability of successful key exfiltration against $q$.