Public Immunization Against Complete Subversion Without Random Oracles

Giuseppe Ateniese[1], Danilo Francati[1(✉)], Bernardo Magri[2], and Daniele Venturi[3]

[1] Stevens Institute of Technology, Hoboken, NJ, USA
dfrancat@stevens.edu
[2] Department of Computer Science, Aarhus University, Aarhus, Denmark
[3] Department of Computer Science, Sapienza University of Rome, Rome, Italy

Abstract. We seek constructions of general-purpose immunizers that take arbitrary cryptographic primitives, and transform them into ones that withstand a powerful "malicious but proud" adversary, who attempts to break security by possibly subverting the implementation of *all* algorithms (including the immunizer itself!), while trying not to be detected. This question is motivated by the recent evidence of cryptographic schemes being intentionally weakened, or designed together with hidden backdoors, e.g., with the scope of mass surveillance.

Our main result is a subversion-secure immunizer in the plain model (assuming collision-resistant hashing), that works for a fairly *large class* of *deterministic* primitives, i.e., cryptoschemes where a secret (but *tamperable*) random source is used to generate the keys and the public parameters, whereas all other algorithms are deterministic. The immunizer relies on an additional *independent* source of *public* randomness, which is used to sample a public seed. While the public source is *untamperable*, the subversion of all other algorithms is allowed to depend on it.

Previous work in the area only obtained subversion-secure immunization for very restricted classes of primitives, often in weaker models of subversion and relying on random oracles, or by leveraging a higher number of independent random sources.

1 Introduction

A common trend in modern cryptography is to design cryptographic schemes that come with a proof of security in a well-defined model. The proof is typically by reduction, meaning that violating the security of the scheme implies the existence of an efficient algorithm for solving some well-studied mathematical problem which is believed to be hard (e.g., factoring certain integers, or inverting a one-way function). While having such a security proof is a desirable feature, it is at least as important to make sure that the security model fits reality, as otherwise provably secure schemes are of little use in practice.

B. Magri—The author was supported by the Concordium Blockchain Research Center, Aarhus University, Denmark.

R. H. Deng et al. (Eds.): ACNS 2019, LNCS 11464, pp. 465–485, 2019.
https://doi.org/10.1007/978-3-030-21568-2_23

Unfortunately, security models often make idealized assumptions that are not always fulfilled in the real world. In this paper, we focus on one of those gaps, which is the discrepancy between the *specification* of a cryptographic scheme and its *implementation*. In particular, we consider the extreme case where the implementation is fully adversarial, i.e., the adversary is allowed to subvert or substitute some (or possibly all) algorithms in the original specification, with the purpose of weakening security.

The above scenario recently gained momentum due to the NSA leaks by Edward Snowden [3,18,21], and because of the EC_DUAL PRG[1] incident [9]. These hazards challenge modern cryptographers to design protection mechanisms withstanding subversion and tampering, as it was also highlighted by Phil Rogaway in his 2015 IACR Distinguished Lecture [22].

1.1 Background

To guarantee some form of security in such an adversarial setting, we must put some restrictions on the adversary, as otherwise, it is easy to subvert a cryptographic scheme in a way that becomes insecure (e.g., the subverted scheme could always output the secret key). A natural restriction, which is also inspired by real-world attacks, is to demand that subversion should be *undetectable* by honest users. In other words, the adversary's goal is to tamper with the specification of a cryptographic scheme in such a way that the produced outputs appear indistinguishable from that of a faithful implementation, yet they allow an adversary to break security given some additional pieces of information altogether.

As it turns out, the possibility of such attacks was already uncovered more than twenty years ago by Young and Yung [29,30], who dubbed the field kleptography (a.k.a. "cryptography against cryptography"). At Crypto 2014, Bellare, Paterson, and Rogaway [7] revisited this setting for the concrete case of symmetric encryption. In particular, on the one hand, they showed that it is possible to hide a backdoor in the encryption algorithm of any sufficiently randomized symmetric encryption scheme in such a way that the produced ciphertexts appear indistinguishable from honestly computed ones, yet knowledge of the backdoor allows the adversary to extract the secret key in full; on the other hand, they suggested that deterministic symmetric encryption schemes are secure against all subversion attacks that meet some form of undetectability. Their results were later extended in several ways [6,10], while follow-up work studied similar questions for the case of digital signatures [1], pseudorandom generators [11,12], non-interactive zero knowledge [5], key encapsulation [2], and hash functions [16,25].

Complete Subversion. A common feature of the works above is that only some of the algorithms underlying a given cryptographic scheme are subject to subversion, while the others are assumed to follow the original specification faithfully.

[1] The PRG was standardized by NIST in 2006, and later withdrawn in 2014 as it was including a potential backdoor allowing to predict future outputs of the PRG algorithm.

Motivated by this limitation, Russell *et al.* [23] put forward a new framework where the adversary is allowed to subvert *all* algorithms; furthermore, in order to cast undetectability, they introduced a trusted third party, a so-called watchdog, whose goal is to test whether the (possibly subverted) implementation is compliant with the original specification of a cryptographic scheme. In a nutshell, a primitive is subversion secure if there exists a universal watchdog such that either no adversary subverting all algorithms can break the security of the scheme, or, if instead, a subversion attack is successful, the watchdog can detect it with non-negligible probability.

The testing procedure executed by the watchdog is typically performed only once before the (possibly subverted) scheme is used "in the wild". This is known as the *offline* watchdog model. Unfortunately, there are subversion attacks that cannot be detected in an offline fashion. Think, e.g., of a signature scheme where the signature algorithm is identical to the original specification, except that upon input of a special message (that is also hard-wired in the implementation) it compromises security (e.g., it returns the secret key). Now, assuming that the message space is large enough, an offline watchdog has a negligible chance of hitting this hidden trigger, so that the subverted implementation will pass the test phase; yet, the subverted scheme is clearly insecure (in the standard sense of unforgeability against chosen-message attacks).

To cast such attacks, [23] introduces the *online* watchdog model, where the watchdog is essentially allowed to additionally monitor the public interaction between users while the scheme is being used "in the wild" (on top of performing the same offline testing, as before).[2]

Cliptography. The main contribution of Russell *et al.* [23], apart from introducing the model of complete subversion, is to propose a methodology to clip the power of subversion attacks against one-way (trapdoor) permutations. Moreover, they show how to rely on such subversion-secure one-way permutations to derive subversion-secure pseudorandom generators and digital signatures. All their results are in the random oracle model (ROM) of Bellare and Rogaway [8].

In a follow-up paper [24], the same authors show how to obtain public-key chosen-plaintext attack secure encryption resisting complete subversion, again in the ROM. This result (inherently) requires the assumption of two *independent* secret, but tamperable, sources of randomness. They further show that their construction can be instantiated in the standard model (i.e., without random oracles) assuming a super-constant number of *independent* sources.

Open Questions. The works of [23,24] only cover a limited set of cryptographic primitives. Furthermore, the assumption of having a large number of *independent* sources is quite a strong one in practice [28]. Hence, the natural question:

[2] One can imagine even more powerful watchdogs monitoring public transcripts while being given the user's secret keys; these are known as *omniscent* watchdogs, but will not be considered in this paper.

Is it possible to protect other primitives against complete subversion, by relying on a single source of secret, but tamperable, randomness, and without assuming random oracles?

1.2 Our Contributions

In this paper, we make significant progress towards answering the above question. Our starting point is a notion of subversion-resistant immunizer Ψ, whose goal is to take an arbitrary primitive Π that is secure w.r.t. some game $\mathbf{G}$, and transform it into an immunized primitive $\Pi^* = \Psi(\Pi)$ (for the same cryptographic task) that is secure w.r.t. $\mathbf{G}$ under complete subversion (in the sense of [23]). The immunizer leverages two *independent* random sources, which we denote by R and S: The source R is an m-bit source which is assumed to be *secret*, but tamperable; the source S is an ℓ-bit source which is assumed to be *public* but *untamperable*. The subversion $\widetilde{\Pi}$ is allowed to depend on the seed s sampled from S and used by the immunized cryptosystem (i.e., first s is sampled and made public, and then the adversary subverts Π^*).

Next, we show how to construct a subversion-secure immunizer tailored to protect *deterministic* primitives Π (secure w.r.t. some game $\mathbf{G}$), where the latter means that the original specification of Π consists of a secret random m-bit source R that is sampled in order to generate the public/secret keys of the scheme (via an algorithm K), and the public parameters (via an algorithm P), whereas every other algorithm F_i underlying Π is deterministic. Our immunizer can be instantiated using any collision-resistant hash function, but for certain primitives Π two additional properties are required (more on this later).

Interestingly, our results allow us to protect new cryptographic primitives against complete subversion; examples include: (weak) pseudorandom functions and permutations, message authentication codes, collision/second pre-image/pre-image resistant hash functions, deterministic symmetric encryption, and more. Previously to our work, for the primitives mentioned above, it was only known how to obtain security in weaker models of subversion, or with random oracles. We refer the reader to Table 1 for a comparison of our results with state-of-the-art research in the area.

1.3 Techniques

We turn to a high-level description of the techniques behind our results. Let $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ be a *deterministic* cryptographic scheme. As explained above, algorithms P and K are responsible to generate, respectively, global public parameters ρ and a public/secret key pair (pk, sk) that are taken as input by all other algorithms F_i.[3] Importantly, all algorithms are deterministic, except for P and K which further take as input independent random coins $r \in \{0,1\}^m$ generated by sampling a secret, uniformly random, source R.

[3] The string pk might be empty for secret-key primitives.

Table 1. Comparing our constructions with other results for security under subversion. We use the following abbreviations: "Pub" for public, "Sec" for secret, "CPA-SKE/CPA-PKE" for public/secret-key encryption under chosen-plaintext attacks, "PRG" for pseudorandom generator, "OWF/TDF" for one-way (trapdoor) function, "CRH" for collision-resistant hash function, "ROM" for random oracle model, "$\forall$ det-unp" for all deterministic primitives with security w.r.t. an unpredictability game, "$\forall$ det-ind^2" for all deterministic primitives with *square* security w.r.t. an indistinguishability game. The value δ is a small constant. The green color means the source is assumed to be untamperable.

Reference	Primitive	Complete Subversion	# of Sources		Additional Assumptions	Notes
			Pub	Sec		
[7]	CPA-SKE	✗	0	1	–	Unique ciphertexts
[1]	SIG	✗	0	1	–	Unique signatures
[12]	PRG	✗	1	1	ROM	–
[23]	OWF/TDF	✓	0	1	ROM	–
	PRG	✓	0	1	ROM	–
	SIG	✓	0	1	ROM	–
[24]	CPA-PKE	✓	0	2	ROM	–
	CPA-PKE	✓	0	$O(\lambda^\delta/\log\lambda)$	OWF	–
§4	$\forall$ det-unp	✓	1	1	CRH	Public model, Single Instance
	$\forall$ det-ind^2	✓	1	1	CRH	

Our immunization strategy follows the design principle of "decomposition and trusted amalgamation" introduced in [24], by means of hash functions $h_{s_1}, h_{s_2} : \{0,1\}^n \to \{0,1\}^m$ with seeds s_1, s_2 sampled independently from a public source S. More in details, we take $2k \stackrel{\text{def}}{=} 2n/m$ samples $r_1^1, \ldots, r_k^1$ and $r_1^2, \ldots, r_k^2$ from the (possibly subverted) source R, and then we hash the amalgamated strings $r_1 \stackrel{\text{def}}{=} r_1^1 || \cdots || r_k^1$ and $r_2 \stackrel{\text{def}}{=} r_1^2 || \cdots || r_k^2$, respectively, using seeds s_1 and s_2. Finally, the immunized parameter generation algorithm P^* runs $\mathsf{P}(1^\lambda; h_{s_1}(r_1))$, whereas the immunized key generation algorithm K^* runs $\mathsf{K}(1^\lambda; h_{s_2}(r_2))$; the algorithms $(\mathsf{F}_i)_{i \in N}$ are not modified.

Intuitively, the above immunizer tries to sanitize the randomness used for parameters/keys generation in such a way that it is harder for an adversary to generate such values together with a backdoor. We stress that the trick of hashing the random coins for key generation was introduced by [23], although there it was applied only to immunize trapdoor permutations in the ROM, whereas we generalize their approach in such a way that it can be applied to a large class of deterministic primitives (as defined above) in the plain model.

Input Constrained/Unconstrained Games. Recall that for some primitives it is inherently impossible to obtain subversion security in the offline watchdog model. Hence, in our analysis of the above immunizer, we identify a natural property of cryptographic games which allows us to prove security in the *offline* watchdog model; for games not satisfying this property we instead obtain security in the *online* watchdog model.

More in details, a game $\mathbf{G}$ for some primitive Π consists of an interaction between an adversary A and a challenger C, where C is given oracle access to

the algorithms underlying Π in order to answer queries from A, and determine whether A wins the game or not. We call $\mathbf{G}$ *input constrained*, if the inputs x_i upon which each (deterministic) algorithm F_i is queried during the game are sampled by C via some public distribution D_i that is independent of the adversary. On the other hand, a game that is not input constrained is called *input unconstrained*. Examples of input-constrained games $\mathbf{G}$ include, e.g., the standard security games for weak pseudorandom functions and one-way permutations. See Sect. 2.2 for more examples.

Security Proof. We prove security of the above immunizer assuming the hash functions h_{s_1}, h_{s_2} are min-entropy condensers for seed-dependent sources. Intuitively, this means that given a uniform ℓ-bit seed s and an n-bit input x coming from a possibly adversarial (but efficiently sampleable) source which might depend on s, and with min-entropy at least k, the output $h_s(x)$ is an m-bit string whose distribution is computationally close to that of an efficiently sampleable source Y with min-entropy at least $m - d$. Such condensers were constructed by Dodis *et al.* [14] using sufficiently strong collision-resistant hash functions.

Fix some primitive Π with input-constrained game $\mathbf{G}$. Let us start with the original subversion game, where first the seeds s_1, s_2 are sampled (from the untamperable public source S) and given to the adversary. Then, the attacker specifies a subversion $\widetilde{\Pi}$ for the immunized cryptosystem; hence, the adversary interacts with the challenger, which first samples random strings $r_1 = r_1^1||\cdots||r_k^1$ and $r_2 = r_1^2||\cdots||r_k^2$, using the subverted source $\widetilde{\mathsf{R}}$ as explained above, and then plays the game $\mathbf{G}$ for Π, given oracle access to the subverted algorithms $\widetilde{\mathsf{P}}, \widetilde{\mathsf{K}}, (\widetilde{\mathsf{F}}_i)_{i\in[N]}$. By contradiction, assume that there is an adversary A that wins the subversion game, but for which no watchdog W can detect the subversion. We then proceed with a sequence of hybrids, as outlined below:

1. In the 1st hybrid, we replace algorithms $\widetilde{\mathsf{K}}$, $\widetilde{\mathsf{P}}$, and $\widetilde{\mathsf{F}}_i$, with their genuine immunized implementation $\mathsf{K}^*(1^\lambda;\cdot) = \mathsf{K}(1^\lambda; h_{s_1}(\cdot))$, $\mathsf{P}^*(1^\lambda;\cdot) = \mathsf{P}(1^\lambda; h_{s_2}(\cdot))$, and $(\mathsf{F}_i^*)_{i\in[N]} = (\mathsf{F}_i)_{i\in[N]}$. One can show that any distinguisher between the original game and this hybrid can be turned into an efficient offline watchdog W detecting the subversion of A. Thus, the two experiments are computationally close.
2. In the 2nd hybrid, we now generate the public parameters and the keys by running $\mathsf{P}(1^\lambda; y_1)$ and $\mathsf{K}(1^\lambda; y_2)$, where y_1, y_2 come from the source Y guaranteed by the condenser. To argue indistinguishability, assume for simplicity that the subverted source $\widetilde{\mathsf{R}}$ is stateless.[4] First, we show that $\widetilde{\mathsf{R}}$ has a non-trivial amount of min-entropy, as otherwise, it is again possible to construct a watchdog W that detects subversion. Second, we argue that since $\widetilde{\mathsf{R}}$ is stateless and efficiently sampleable, the strings $r_1 = r_1^1||\cdots||r_k^1$ and $r_2 = r_1^2||\cdots||r_k^2$ have min-entropy at least k, so that indistinguishability of the two experiments follows by security of the min-entropy condenser. Note that the last

[4] The case of stateful subversion can be reduced to that of stateless subversion if we assume that watchdogs are allowed to reset the state of a tested implementation, a trick due to [23].

step is possible because the public random source S is untamperable, and moreover, the subverted random source $\widetilde{\mathsf{R}}$ has non-trivial min-entropy even conditioned on s_1, s_2 sampled from S.

3. Finally, in order to conclude the proof, we exploit the framework of "overcoming weak expectations" by Dodis and Yu [15], who established that for a *large class* of primitives[5] there is a natural trade-off between concrete security and the capacity to withstand a certain entropy deficiency d on the distribution of the key A technical challenge here comes from the fact that this framework only applies to cryptosystems Π where the secret key is uniformly random (and moreover there are no public parameters, or those are generated using uniform randomness). However, we show a similar tradeoff still holds for our specific setting, at least for *single-instance* games where the original random source R is sampled only twice (one for generating the public parameters, and one for sampling the keys).[6]

1.4 Comparison with Russell *et al.* [23,24]

The trick of splitting a cryptographic algorithm into several sub-components (as we do for $\mathsf{P}, \mathsf{K}, \mathsf{R}$) was originally introduced in [23], and later refined in [24], under the name of "split-program" methodology. Remarkably, [24] shows that for semantically-secure public-key encryption (an inherently *randomized* primitive) de-coupling the encryption algorithm in a randomized component R (for generating the random coins) and a deterministic component Enc (for computing the ciphertext) is not sufficient to defeat kleptographic attacks. For this reason, they propose a "double-splitting" technique where R is further split into two (tamperable) components $\mathsf{R}_1, \mathsf{R}_2$. In this perspective, our immunization strategy can be thought of as a form of "double splitting", where one of the two sources is assumed to be untamperable but made *public*.

The fact that subversion-secure immunization in the offline watchdog model only works for input-constrained games is reminiscent of a general observation made in [23] stating that an offline watchdog can always detect the subversion of deterministic algorithms with public input distributions (see [23, Lemma 2.3]).

Finally, we would like to stress that our work only covers immunization against complete subversion in the form of algorithm-substitution attacks. In particular, the adversary always specifies an algorithm $\widetilde{\mathsf{P}}$ that is used for sampling the public parameters during the security game. Hence, our immunizers do not provide any guarantee in the "adversarially chosen parameters model" considered in [11,12,16,23] (where the adversaries specify the malicious public parameters directly).

[5] In particular, the result of [15] applies to all unpredictability primitives, and all indistinguishability primitives meeting so-called *square* security.

[6] Hence, our results do not cover, e.g., multi-instance games where several public parameters and keys might be generated.

1.5 Further Related Work

The original attacks in the kleptographic setting extended previous work on subliminal channels by Simmons [26,27]. This research is also intimately connected to the problem of steganography, whose goal in the context of secret communication is to hide the mere fact that messages are being exchanged [19].

Dodis *et al.* [12], study different immunization strategies for backdoored pseudorandom generators. While they do not consider complete subversion, as the immunizer and the PRG algorithm are assumed to be trusted, they deal with the case where a cryptographic scheme might be subverted "by design" (e.g., because it is standardized with maliciously generated public parameters).

Another line of work suggests defeating subversion attacks employing a cryptographic reverse firewall [1,13,20]. Such a firewall is used to re-randomize the incoming/outgoing messages of a potentially subverted primitive. The firewall itself is assumed to be trusted, and moreover, it relies on a secret, and untamperable, random source. Yet another approach consists of designing self-guarding schemes [17], which allow us to defeat subversion without relying on external parties (such as watchdogs or reverse firewalls), at the price of assuming a secure initialization phase where the primitive to protect was not under subversion.

2 Preliminaries

2.1 Notation

We use the notation $[n] \stackrel{\text{def}}{=} \{1, \ldots, n\}$. Capital letters (such as X) are used to denote random variables, caligraphic letters (such as $\mathcal{X}$) to denote sets, and sans serif letters (such as A) to denote algorithms. All algorithms in this paper are modelled as (possibly interactive) Turing machines.

For a string $x \in \{0,1\}^*$, we let $|x|$ be its length; if $\mathcal{X}$ is a set, $|\mathcal{X}|$ represents the number of elements in $\mathcal{X}$. When x is chosen randomly in $\mathcal{X}$, we write $x \xleftarrow{\$} \mathcal{X}$. If A is an algorithm, we write $y \xleftarrow{\$} \mathsf{A}(x)$ to denote a run of A on input x and output y; if A is randomized, then y is a random variable and $\mathsf{A}(x;r)$ denotes a run of A on input x and (uniform) randomness r. An algorithm A is *probabilistic polynomial-time* (PPT) if A is randomized and for any input $x, r \in \{0,1\}^*$ the computation of $\mathsf{A}(x;r)$ terminates in a polynomial number of steps (in the size of the input). We denote the expected value of a random variable X as $\mathbb{E}[X]$.

Negligible Functions. Throughout the paper, we denote by $\lambda \in \mathbb{N}$ the security parameter. A function $\nu : \mathbb{N} \to [0,1]$ is called *negligible* in the security parameter λ if it vanishes faster than the inverse of any polynomial in λ, i.e. $\nu(\lambda) \in O(1/p(\lambda))$ for all positive polynomials $p(\lambda)$. We sometimes write $\mathsf{negl}(\lambda)$ (resp., $\mathsf{poly}(\lambda)$) to denote all negligiblie functions (resp., polynomial functions) in the security parameter.

Unpredictability and Indistinguishability. The min-entropy of a random variable $X \in \mathcal{X}$ is $\mathbb{H}_\infty(X) \stackrel{\text{def}}{=} -\log \max_{x \in \mathcal{X}} \mathbb{P}[X = x]$, and intuitively it measures the best chance to predict X (by a computationally unbounded algorithm). For conditional distributions, unpredictability is measured by the conditional average min-entropy $\widetilde{\mathbb{H}}_\infty(X|Y) \stackrel{\text{def}}{=} -\log \mathbb{E}_y \left[2^{-\mathbb{H}_\infty(X|Y=y)}\right]$.

The statistical distance between two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, is defined as $\mathbb{SD}(X;Y) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{v \in \mathcal{X} \cup \mathcal{Y}} |\mathbb{P}[X = v] - \mathbb{P}[Y = v]|$. Let $X = \{X_\lambda\}_{\lambda \in \mathbb{N}}$ and $Y = \{Y_\lambda\}_{\lambda \in \mathbb{N}}$ be two ensembles of random variables. We say that X and Y are *statistically* indistinguishable, denoted $X \approx_s Y$, as a shortening for $\mathbb{SD}(X_\lambda; Y_\lambda) \in \mathsf{negl}(\lambda)$. Similarly, we say that X and Y are *computationally* indistinguishable, denoted $X \approx_c Y$, if for all PPT distinguishers D we have $\Delta_{\mathsf{D}}(X_\lambda; Y_\lambda) \in \mathsf{negl}(\lambda)$, where

$$\Delta_{\mathsf{D}}(X_\lambda; Y_\lambda) \stackrel{\text{def}}{=} \left|\mathbb{P}\left[\mathsf{D}(1^\lambda, X_\lambda) = 1\right] - \mathbb{P}\left[\mathsf{D}(1^\lambda, Y_\lambda) = 1\right]\right|.$$

An ensemble $X = \{X_\lambda\}_{\lambda \in \mathbb{N}}$ is efficiently sampleable if there exists a PPT algorithm X such that, for each $\lambda \in \mathbb{N}$, the output of $\mathsf{X}(1^\lambda)$ is distributed identically to X_λ.

2.2 Abstract Games

In this work, we deal with abstract cryptographic schemes. Usually, a cryptographic scheme is just a sequence of (possibly randomized) efficient algorithms. However, for our purpose, it will be convenient to specify two special algorithms which are common to any cryptographic scheme; those are the algorithms for generating the public/secret keys and the public parameters (if any). Moreover, our focus will be on *deterministic* schemes (see below).

In this vein, a deterministic cryptographic scheme is a sequence of efficient algorithms $\Pi \stackrel{\text{def}}{=} (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$, where:

- P is a deterministic algorithm that upon input the security parameter 1^λ, and random coins $r \in \mathcal{R}$, outputs public parameters $\rho \in \mathcal{P}$;
- K is a deterministic algorithm that upon input the security parameter 1^λ, and random coins $r \in \mathcal{R}$,[7] outputs a pair of keys $(pk, sk) \in \mathcal{PK} \times \mathcal{SK}$;
- The random coins for (P, K) are obtained via independent calls to algorithm R, which outputs a uniformly random string $r \in \mathcal{R}$ upon each invocation.
- For each $i \in [N]$, algorithm $\mathsf{F}_i : \mathcal{X}_i \to \mathcal{Y}_i$ is deterministic.

We stress that the above syntax is meant to capture both secret-key and public-key primitives; in the former case the public key is simply equal to the empty string $pk = \varepsilon$, and $\mathcal{PK} = \emptyset$. Further, without loss of generality, we assume that all algorithms $\mathsf{F}_1, \ldots, \mathsf{F}_N$ take as input both ρ and (pk, sk); the key generation algorithm also receives ρ as additional input.

[7] We assume the amount of randomness to generate the public parameters and the keys is the same; a generalization is straightforward.

Typically, a cryptographic scheme must meet two properties. The first is a *correctness* requirement, which essentially says that Π correctly implements the desired functionality;[8] although we will not define correctness in general, we will later assume Π meets some well-defined correctness property. The second is a *security* requirement, which we model as an interactive process (a.k.a. game) between an adversary and a challenger.

Definition 1 (Cryptographic game). *A cryptographic game* $\mathbf{G} \stackrel{def}{=} (\mathsf{C}, \gamma)$ *is defined by a challenger* C *and a constant* $\gamma \in [0,1)$*; the game is (implicitly) parametrized by a cryptographic scheme* $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$*, an adversary* A*, and the security parameter* $\lambda \in \mathbb{N}$*. In an execution of the game the (efficient) challenger* $\mathsf{C}(1^\lambda)$ *interacts with the (efficient) adversary* $\mathsf{A}(1^\lambda)$*, and at the end the challenger outputs a decision bit* $d \in \{0,1\}$*. We denote the output of the game as* $d \xleftarrow{\$} \langle \mathsf{A}(1^\lambda), \mathsf{C}^{\mathsf{P},\mathsf{K},\mathsf{R},(\mathsf{F}_i)_{i\in[N]}}(1^\lambda)\rangle$*; we sometimes also write* $(d,\tau) \xleftarrow{\$} (\mathsf{A}(1^\lambda) \leftrightarrows \mathsf{C}^{\mathsf{P},\mathsf{K},\mathsf{R},(\mathsf{F}_i)_{i\in[N]}}(1^\lambda))$ *for a transcript of the interaction between the adversary and the challenger,* C^{Π} *as a shorthand for* $\mathsf{C}^{\mathsf{P},\mathsf{K},\mathsf{R},(\mathsf{F}_i)_{i\in[N]}}$*, and* $\mathbf{G}_{\Pi,\mathsf{A},\mathsf{C}}$ *for the random variable corresponding to an execution of game* $\mathbf{G}$ *with scheme* Π*, adversary* A*, and challenger* C*.*

We say that Π *is* (t,ϵ)-secure *w.r.t. game* $\mathbf{G} = (\mathsf{C}, \gamma)$ *if the following holds: For all probabilistic attackers* A *running in time* t *we have*

$$\left| \mathbb{P}\left[d = 1 : \; d \xleftarrow{\$} \langle \mathsf{A}(1^\lambda), \mathsf{C}^{\mathsf{P},\mathsf{K},\mathsf{R},(\mathsf{F}_i)_{i\in[N]}}(1^\lambda)\rangle \right] - \gamma \right| \leq \epsilon.$$

Moreover, whenever for all $t \in \mathsf{poly}(\lambda)$ *there exists* $\epsilon \in \mathsf{negl}(\lambda)$ *such that* Π *is* (t,ϵ)*-secure w.r.t. game* $\mathbf{G}$*, we simply say that* Π *is secure w.r.t. game* $\mathbf{G}$*.*

Input-Constrained Games. An important distinction will be whether the adversary is allowed or not to choose the inputs for the oracle calls made by the challenger. We call games where the latter is not possible *input-constrained* games.

Definition 2 (Input-constrained games). *Let* $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ *be a cryptographic scheme, and* $\mathbf{G} = (\mathsf{C}, \gamma)$ *be a security game for* Π*. We call* $\mathbf{G}$ input constrained *if the following holds: For each* $i \in [N]$*, there exists a* public *and* efficiently samplable *distribution* D_i*, such that the challenger chooses the inputs to each oracle* F_i *by sampling a fresh and independent value from* D_i*.*

In contrast, games where the above property is not met are called *input unconstrained*. We provide a few clarifying examples below.

One-Way Functions: A one-way function (OWF) is a cryptographic scheme $\Pi = (\mathsf{P}, \mathsf{R}, \mathsf{OWF})$ where $N = 1$, and $\mathsf{OWF} : \mathcal{X} \to \mathcal{Y}$ is a function. Security of Π is characterized by a game $\mathbf{G}^{\mathrm{owf}} = (\mathsf{C}_{\mathrm{owf}}, 0)$ defined as follows: (i) $\mathsf{C}_{\mathrm{owf}}$ picks $\rho = \mathsf{P}(1^\lambda; r)$ (for uniform $r \xleftarrow{\$} \mathsf{R}(1^\lambda)$), samples $x \xleftarrow{\$} \mathcal{X}$, computes

[8] For instance, if Π is a signature scheme, correctness demands that honestly computed signatures (w.r.t. a valid secret key) always verify correctly (w.r.t. the corresponding public key).

$y = \mathsf{OWF}(1^\lambda, \rho, x)$, and sends (ρ, y) to the adversary; (ii) A wins iff it returns a values $x' \in \mathcal{X}$ such that $\mathsf{OWF}(1^\lambda, \rho, x') = y$. Notice that C_{owf} needs to invoke oracle OWF upon input x' in order to determine the decision bit d, and thus the game is input unconstrained.

One-Way Permutations: A one-way permutation (OWP) is a cryptographic scheme $\Pi = (\mathsf{P}, \mathsf{R}, \mathsf{OWP})$ where $N = 1$, and $\mathsf{OWP} : \mathcal{X} \to \mathcal{X}$ is a permutation. Security of Π is characterized by a game $\mathbf{G}^{\text{owp}} = (\mathsf{C}_{\text{owp}}, 0)$ defined as follows: (i) C_{owp} picks $\rho = \mathsf{P}(1^\lambda; r)$ (for uniform $r \xleftarrow{\$} \mathsf{R}(1^\lambda)$), samples $x \xleftarrow{\$} \mathcal{X}$, computes $y = \mathsf{OWP}(1^\lambda, \rho, x)$, and sends (ρ, y) to the adversary; (ii) A wins iff it returns a value $x' \in \mathcal{X}$ such that $x' = x$. Notice that C_{owp} does not need to make any oracle call in order to determine the decision bit d, and thus the game is input constrained with public distribution D equal to the uniform distribution over the domain $\mathcal{X}$.

(Weak) Pseudorandom Functions: A pseudorandom function (PRF) is a cryptographic scheme $\Pi = (\mathsf{P}, \mathsf{R}, \mathsf{R}, \mathsf{PRF})$ where $N = 1$, and $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a keyed function. Security of Π is characterized by a game $\mathbf{G}^{\text{prf}} = (\mathsf{C}_{\text{prf}}, 1/2)$ defined as follows: (i) C_{prf} samples a bit $b \xleftarrow{\$} \{0,1\}$, picks $\rho = \mathsf{P}(1^\lambda; r_1)$ and $\kappa = \mathsf{K}(1^\lambda, \rho; r_2)$ (where $r_1, r_2 \xleftarrow{\$} \mathsf{R}(1^\lambda)$), and sends ρ to the adversary; (ii) A can ask queries of the form $x \in \mathcal{X}$, upon which C_{prf} either replies with $y = \mathsf{PRF}(\kappa, x)$ (in case $b = 0$) or $y \xleftarrow{\$} \mathcal{Y}$ (in case $b = 1$); (iii) A returns a bit b' and wins iff $b = b'$. Notice that C_{prf} needs to invoke oracle PRF upon inputs specified by the adversary, and thus the game is input unconstrained.

For *weak* PRFs the game is changed as follows: In step (ii) the queries made by the adversary are empty, and instead C_{prf} samples $x \xleftarrow{\$} \mathcal{X}$ and returns (x, y), where y is computed as before. Hence, the game is constrained with public distribution equal to the uniform distribution over $\mathcal{X}$.

Hash Functions: A cryptographic hash function is a cryptographic scheme $\Pi = (\mathsf{P}, \mathsf{R}, \mathsf{Hash})$ where $N = 1$, and $\mathsf{Hash} : \mathcal{X} \to \mathcal{Y}$ is a (typically compressing) function. Security of Π is characterized by a game $\mathbf{G}^{\text{cr}} = (\mathsf{C}_{\text{cr}}, 0)$ defined as follows: (i) C_{cr} picks $\rho = \mathsf{P}(1^\lambda; r)$ (for uniform $r \xleftarrow{\$} \mathsf{R}(1^\lambda)$), and sends ρ to the adversary; (ii) A wins iff it returns a pair of values $(x, x') \in \mathcal{X}^2$ such that $\mathsf{Hash}(1^\lambda, \rho, x) = \mathsf{Hash}(1^\lambda, \rho, x')$ and $x \neq x'$. Notice that C_{cr} needs to invoke oracle Hash upon input x, x' in order to determine the decision bit d, and thus the game is input unconstrained.

Secret-Key Encryption: A deterministic secret-key encryption scheme is a cryptographic scheme $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{Enc}, \mathsf{Dec})$ where $N = 2$. The (deterministic) encryption algorithm takes as input the secret key $\kappa \in \mathcal{K}$ and a message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$. The (deterministic) decryption algorithm takes as input the secret key $\kappa \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$, and outputs a message $m \in \mathcal{M}$ (or an error symbol). Security of a deterministic encryption scheme is characterized, e.g., by a game $\mathbf{G}^{\text{cca-ske}} = (\mathsf{C}_{\text{cca-ske}}, 1/2)$ specified as follows: (i) $\mathsf{C}_{\text{cca-ske}}$ picks $\rho = \mathsf{P}(1^\lambda; r_1)$ and $\kappa = \mathsf{K}(1^\lambda, \rho; r_2)$ (where $r_1, r_2 \xleftarrow{\$} \mathsf{R}(1^\lambda)$), and sends ρ to the adversary; (ii) A can specify encryption queries: Upon input a message $m \in \mathcal{M}$, the

challenger returns $c = \mathsf{Enc}(1^\lambda, \kappa, m)$; (iii) A can specify decryption queries: Upon input a ciphertext $c \in \mathcal{C}$, the challenger returns $m = \mathsf{Dec}(1^\lambda, \kappa, c)$; (iv) A can specify a challenge query: Upon input $(m_0^*, m_1^*) \in \mathcal{M}^2$, the challenger returns $c^* = \mathsf{Enc}(1^\lambda, \kappa, m_b^*)$ where $b \xleftarrow{\$} \{0,1\}$ is a hidden bit; (v) A can continue to specify encryption/decryption queries, with the restriction that c^* cannot be part of a decryption query; (vi) A returns a bit b' and wins iff $b = b'$. Notice that $\mathsf{C}_{\text{cca-ske}}$ needs to invoke oracles $\mathsf{Enc}, \mathsf{Dec}$ in order to answer encryption/decryption queries, and thus the game is input unconstrained.

Single-Instance Games. As mentioned in the introduction, our results only apply to a sub-class of games where the random source R is sampled only twice, in order to obtain the randomness needed for generating the public parameters and the keys. We call such games *single instance*.

Definition 3 (Single-instance games). *Let $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ be a cryptographic scheme, and $\mathbf{G} = (\mathsf{C}, \gamma)$ be a security game for Π. We call* $\mathbf{G}$ single instance *if during a game execution the challenger invokes the oracle R twice, in order to obtain coins r_1, r_2 that are later fed to oracles P, K.*

3 Security Model

In this section, we consider a standard-model definition for subversion security, via so-called *immunizers*. An immunizer is a transformation that takes as input a cryptographic scheme (for some task) and transforms it into another scheme for the same task that withstands complete subversion; the immunizer is allowed to leverage a single source of public, but untamperable, randomness. Importantly, we seek security in the standard model (i.e., without random oracles) and in a setting where the immunizer itself is subject to subversion.

We first define our model formally, in Sect. 3.1, for the case of offline watchdogs. Then, in Sect. 3.2, we discuss some definitional choices and compare our definitions with previous work in the area. In the full version, we explain how to extend our framework to the case of online watchdogs.

3.1 Subversion-Secure Immunizers

Let $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ be a cryptographic scheme (as defined in Sect. 2.2), where we assumed that $\mathcal{R} \stackrel{\text{def}}{=} \{0,1\}^m$ (i.e., the source R is a random m-bit source). An immunizer for Π is a transformation $\Psi[\mathcal{H}, \mathsf{S}]$ parameterized by a family of hash functions $\mathcal{H} = \{h_s : \{0,1\}^n \to \{0,1\}^m\}_{s \in \{0,1\}^\ell}$ and a public random source S over $\{0,1\}^\ell$. We write $\Pi^* \stackrel{\text{def}}{=} \Psi(\Pi) \stackrel{\text{def}}{=} (\mathsf{P}^*, \mathsf{K}^*, \mathsf{R}^*, \mathsf{F}_1^*, \ldots, \mathsf{F}_N^*)$ for the specification of the immunized cryptosystem, where:

- $\mathsf{R}^* \equiv \mathsf{R}$ (i.e., the immunized scheme uses the same secret random source as the original scheme);
- P^* and K^* take as input a seed $s \in \{0,1\}^\ell$, and have n-bit random tapes;

Game $\mathbf{G}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda)$

$s \xleftarrow{\$} \mathsf{S}(1^\lambda)$;

$(\widetilde{\Pi}, \alpha) \xleftarrow{\$} \mathsf{A}_0(1^\lambda, s, \langle \Pi, \Psi \rangle)$

$r_1^1, \ldots, r_k^1, r_1^2, \ldots, r_k^2 \xleftarrow{\$} \widetilde{\mathsf{R}}(1^\lambda)$;

$r_1 = r_1^1 || \cdots || r_k^1; r_2 = r_1^2 || \cdots || r_k^2$;

$(d, \widetilde{\tau}) \xleftarrow{\$} (\mathsf{A}_1(1^\lambda, \alpha) \leftrightarrows \mathsf{C}^{\widetilde{\mathsf{P}}(s,\cdot),\widetilde{\mathsf{K}}(s,\cdot),\widetilde{\mathsf{F}}_1(s,\cdot),\ldots,\widetilde{\mathsf{F}}_N(s,\cdot)}(1^\lambda, r_1, r_2))$

return d

Fig. 1. Games defining subversion security of an immunizer $\Psi[\mathcal{H}, \mathsf{S}]$, in the standard model. We use the notation $\mathsf{C}^{\widetilde{\mathsf{P}}(s,\cdot),\widetilde{\mathsf{K}}(s,\cdot),\widetilde{\mathsf{F}}_1(s,\cdot),\ldots,\widetilde{\mathsf{F}}_N(s,\cdot)}(1^\lambda, r_1, r_2)$ to denote a run of the challenger C with random coins r_1, r_2 (that will be used as input of algorithms $\widetilde{\mathsf{P}}, \widetilde{\mathsf{K}}$ during the game).

- $(\mathsf{F}_i^*)_{i \in N}$ take as input a seed $s \in \{0,1\}^\ell$ plus the same inputs as the corresponding algorithm in Π;
- The seed s is obtained by sampling the public random source S (i.e., $s \xleftarrow{\$} \mathsf{S}(1^\lambda)$).

We require an immunizer Ψ to satisfy two properties. The first property is the usual correctness requirement, meaning that the immunized primitive Π^* meets the same correctness condition as that of Π (for every possible choice of the seed for the hash function). The second property is some flavor of security to subversion attacks. More in details, the public source S is assumed to be untamperable and uniform. The adversary A knows a description of the immunizer Ψ and of the original primitive Π, and is allowed to choose $\widetilde{\Pi} = (\widetilde{\mathsf{P}}, \widetilde{\mathsf{K}}, \widetilde{\mathsf{R}}, (\widetilde{\mathsf{F}}_i)_{i \in N})$ depending on the actual seed $s \in \{0,1\}^\ell$ that is sampled from the public source S during a trusted setup phase (which might be run by an external party). Finally, the adversary plays the security game for Π, where the challenger picks $2n/m := 2k$ samples $(r_i^1, r_i^2)_{i \in [k]}$ from $\widetilde{\mathsf{R}}$, amalgamates them into strings $r_1 = r_1^1 || \cdots || r_k^1$ and $r_2 = r_1^2 || \cdots || r_k^2$, and finally interacts with A given black-box access to $\widetilde{\mathsf{P}}(s,\cdot), \widetilde{\mathsf{K}}(s,\cdot), \widetilde{\mathsf{F}}_i(s,\cdot)$ (i.e., to the subversion specified by the adversary using seed $s \in \{0,1\}^\ell$), where r_1 and r_2 are used as inputs for $\widetilde{\mathsf{P}}$ and $\widetilde{\mathsf{K}}$, respectively. Note that $\widetilde{\Pi}$ is completely arbitrary, and thus all algorithms (including the immunizer) are subject to subversion.

We define the advantage of adversary A in the subversion game with primitive Π, immunizer Ψ, and challenger C as:

$$\mathbf{Adv}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda) \stackrel{\mathrm{def}}{=} \left| \mathbb{P}\left[\mathbf{G}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda) = 1 \right] - \gamma \right|, \tag{1}$$

where the game $\mathbf{G}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda)$ is depicted in Fig. 1, and the probability is taken over the randomness of $\widetilde{\mathsf{S}}, \widetilde{\mathsf{R}}, \mathsf{S}, \mathsf{R}$, and over the coin tosses of A.

Game $\mathbf{G}^{\text{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda,\mathsf{aux},b)$	$\mathbf{G}^{\text{det-on}}_{\Pi,\Psi,\mathsf{W}}(\lambda,\mathsf{aux},b)$
$\mathsf{aux} \stackrel{\text{def}}{=} (\langle\widetilde{\Pi}\rangle, s)$	$\mathsf{aux} \stackrel{\text{def}}{=} (\langle\widetilde{\Pi}\rangle, s, \widetilde{\tau})$
if $b = 0$	**if** $b = 0$
return $\mathsf{W}^{\widetilde{\Pi}}(1^\lambda, \langle\Pi,\Psi\rangle, s)$	**return** $\mathsf{W}^{\widetilde{\Pi}}(1^\lambda, \langle\Pi,\Psi\rangle, s, \widetilde{\tau})$
elseif $b = 1$	**elseif** $b = 1$
$\Pi^* = \Psi(\Pi)$	$\Pi^* = \Psi(\Pi)$
return $\mathsf{W}^{\Pi^*}(1^\lambda, \langle\Pi,\Psi\rangle, s)$	**return return** $\mathsf{W}^{\Pi^*}(1^\lambda, \langle\Pi,\Psi\rangle, s, \widetilde{\tau})$
fi	**fi**

Fig. 2. Description of the detection game of an immunizer $\Psi[\mathcal{H},\mathsf{S}]$ with offline (left) and online (right) watchdogs, in the standard model. The auxiliary information aux is taken from the subversion game (cf. Fig. 1).

Clearly, since the subverted cryptosystem $\widetilde{\Pi}$ specified by the adversary is completely arbitrary, it might be trivial to break security in the above setting. (E.g., consider Π to be a signature scheme and the corresponding subversion to have the signing algorithm return the signing key.) Hence, we need to restrict the adversary in some way. Following previous work, we will consider the adversary to be "malicious-but-proud" in the sense that in order to be successful a subversion attack should also be *undetectable* by the honest user. The latter is formalized by a detection game featuring an efficient algorithm, called the watchdog, whose goal is to detect whether a subversion took place. In particular, given a description of the immunizer and the original scheme, the watchdog has to distinguish the immunized cryptosystem Π^* from the subversion $\widetilde{\Pi}$ used by the adversary in the subversion game. The detect advantage of watchdog W is defined as:[9]

$$\mathbf{Adv}^{\text{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda) \stackrel{\text{def}}{=} \left|\mathbb{P}\left[\mathbf{G}^{\text{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda,\mathsf{aux},0)=1\right] - \mathbb{P}\left[\mathbf{G}^{\text{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda,\mathsf{aux},1)=1\right]\right|, \quad (2)$$

where the game $\mathbf{G}^{\text{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda,\mathsf{aux},b)$ is depicted in Fig. 2, and the probability is taken over the randomness of $\widetilde{\mathsf{S}}, \widetilde{\mathsf{R}}, \mathsf{S}, \mathsf{R}$, and over the coin tosses of W; the values in the auxiliary information aux are taken from $\mathbf{G}^{\text{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda)$. Similarly to previous work, we assume that W has rewinding black-box access to its oracles, a feature required in order to detect stateful subversion [23, Remark 2.5].
We are now ready to define subversion security of an immunizer for the offline watchdog.

[9] Of course, we could also treat the detection game as an indistinguishability game $\mathbf{G} = (\mathsf{C},\gamma)$, and thus define the detection advantage as a function of $\gamma = 1/2$. However, we prefer the above formulation in order to be consistent with previous work [23,24].

Definition 4 (Subversion-resistant immunizer). *Let* $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ *be a cryptographic scheme, and* $\mathbf{G} = (\mathsf{C}, \gamma)$ *be a security game for* Π*. For a constant* $c^* \geq 1$*, and a family of hash functions* $\mathcal{H} = \{h_s : \{0,1\}^n \to \{0,1\}^m\}_{s \in \{0,1\}^\ell}$*, we say that an immunizer* $\Psi[\mathcal{H}, \mathsf{S}]$ *is* $(t_\mathsf{A}, t_\mathsf{W}, c^*, \epsilon^*)$*-subversion-resistant with an* offline *watchdog if the following holds: There exists a watchdog* W *with running time* t_W *such that for all adversaries* A *with running time* t_A *for which* $\mathbf{Adv}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda) > \epsilon^*$*, we have*

$$\mathbf{Adv}^{\mathrm{det}}_{\Pi,\Psi,\mathsf{W}}(\lambda) \geq \frac{1}{c^*} \cdot \mathbf{Adv}^{\mathrm{pub}}_{\Pi,\Psi,\mathsf{A},\mathsf{C}}(\lambda).$$

Moreover, for all $s \in \{0,1\}^\ell$*, we require that the immunized cryptosystem with seed* s *meets the same correctness requirement as that of* Π*.*

Remark 1 (On subverting the immunizer). We stress that the subversion $\widetilde{\Pi}$ should be thought of as the subversion of the *immunized* cryptosystem $\Pi^* = \Psi(\Pi)$. In particular, since the subversion is completely arbitrary, the latter means that the adversary can tamper with (and, in fact, completely bypass) the immunizer itself.

Remark 2 (On including the seed in the auxiliary information). Note that the seed s sampled during the subversion game is part of the auxiliary information aux, and later given as additional input to the watchdog in the detection game.

It is easy to see that the latter is necessary. Consider, for instance, a signature scheme $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{Sign}, \mathsf{Vrfy})$, and let $\Pi^* = (\mathsf{P}^*, \mathsf{K}^*, \mathsf{R}^*, \mathsf{Sign}^*, \mathsf{Vrfy}^*) = \Psi(\Pi)$ be the immunized version of Π. Since the subversion $\widetilde{\Pi}$ is allowed to depend on the seed s, the adversary could instruct $\widetilde{\mathsf{K}}$ to output a fixed verification/signature key pair $(\overline{vk}, \overline{sk})$, known to the adversary, whenever $\widetilde{\mathsf{K}}$ is run upon input s. Now, if the watchdog W would not be given as input the actual seed s, the above attack would be undetectable, as W has only a negligible chance of hitting the seed s while sampling the source S.

3.2 Discussion

On rough terms, Definition 4 says the following. There exists a universal (efficient) watchdog algorithm such that for any adversary that has advantage at least ϵ^* in the subversion game (cf. Eq. (1)), the probability that the watchdog detects the subversion (cf. Eq. (2)) is at least equal to the advantage of the adversary in the subversion game divided by some positive constant $c^* \geq 1$.

We observe that there could be a substantial gap between the value of ϵ^* and the actual advantage of an adversary in the subversion game. In practice, we would like to obtain Definition 4 for small ϵ^*, c^*, such that either the advantage in the subversion game is smaller than ϵ^*, or the advantage in the detection game has a similar magnitude as that in the subversion game (which might be much larger than ϵ^*).

Looking ahead, the choice to state security of immunizers in the style of concrete security will allow us to lower bound the level of unpredictability in the subverted random source $\widetilde{\mathsf{R}}$ with a concrete (rather than asymptotic) value, a feature that will be exploited by our immunizer. One might wonder why Definition 4 considers only a single parameter ϵ^*, instead of having two distinct parameters (i.e., one parameter, say ϵ^*, for the advantage of A in breaking the scheme, and another parameter, say δ^*, for the advantage of W in detecting a subversion). While this might seem like a natural way of phrasing concrete security, it is problematic since such a definition conveys information about a single point over the range of values $\epsilon^*, \delta^* \in [0,1]$. A similar issue was already observed in [10], who also suggested the approach of relating the advantage in the two games.

4 The Immunizer

4.1 Ingredients: Seed-Dependent Randomness Condensers

We recall the notion of seed-dependent randomness condenser [14]. Intuitively, this corresponds to a family of hash functions indexed by an ℓ-bit seed, and mapping n into m bits. The security guarantee is that when the seed s is uniform, and the input x comes from an adversarial, efficiently sampleable, source which might depend on s, and with min-entropy at least k, the output of the hash function has at least $m - d$ bits of min-entropy, for deficiency parameter $d \geq 1$.

Definition 5 (Seed-dependent condenser). *Let $\mathcal{G} \stackrel{def}{=} \{g_s : \{0,1\}^n \to \{0,1\}^m\}_{s\in\{0,1\}^\ell}$ be a family of efficiently computable functions. We say that $\mathcal{G}$ is a family of $(\frac{k}{n} \to \frac{m-d}{m}, t, \epsilon)$-seed-dependent condensers if for all probabilistic adversaries A running in time t who take a seed $s \xleftarrow{\$} \{0,1\}^\ell$ and output (using more coins) a distribution $X \xleftarrow{\$} \mathsf{A}(s)$ of entropy $\widetilde{\mathbb{H}}_\infty(X|S) \geq k$, the joint distribution $(S, g_S(X))$ is ϵ-close to some (S, Y), where $\widetilde{\mathbb{H}}_\infty(Y|S) \geq m - d$ and S is uniform over $\{0,1\}^\ell$.*

4.2 Immunizer Description

We refer the reader to Fig. 3 for a formal description of our immunizer, where we assumed that $\mathcal{R} \stackrel{\text{def}}{=} \{0,1\}^m$. Roughly, the immunizer sanitizes the random coins used to generate the public parameters ρ and the public/secret keys (pk, sk) by first sampling $(r_i^1, r_i^2)_{i\in[k]} \xleftarrow{\$} \mathsf{R}(1^\lambda)$ and amalgamating $r_1 = r_1^1||\cdots||r_k^1$ and $r_2 = r_1^2||\cdots||r_k^2$, and then using, respectively, $h_{s_1}(r_1)$ and $h_{s_2}(r_2)$ as random coins for P and K, where the seeds $s_1, s_2 \in \{0,1\}^\ell$ are sampled using the public source S. All other algorithms are unchanged.

Subversion-Resistant Immunizer $\Psi[\mathcal{H}, \mathsf{S}]$:

Let $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ be a cryptographic scheme, and define $\Pi^* \stackrel{\text{def}}{=} \Psi(\Pi) \stackrel{\text{def}}{=} (\mathsf{P}^*, \mathsf{K}^*, \mathsf{R}^*, \mathsf{F}_1^*, \ldots, \mathsf{F}_N^*)$ as follows.

- Algorithm P^*: Upon input $(1^\lambda, s_1, r_1)$, return $\rho = \mathsf{P}(1^\lambda; h_{s_1}(r_1))$.
- Algorithm R^*: Upon input 1^λ, return r such that $r \xleftarrow{\$} \mathsf{R}(1^\lambda)$.
- Algorithm K^*: Upon input $(1^\lambda, s_2, \rho, r_2)$, return $(pk, sk) = \mathsf{K}(1^\lambda, \rho; h_{s_2}(r_2))$.
- Algorithm F_i^* (for $i \in [N]$): Upon input $(1^\lambda, \rho, (pk, sk), x)$, return $y = \mathsf{F}_i(1^\lambda, \rho, (pk, sk), x)$.

Fig. 3. Description of our subversion-resistant immunizer; the seeds s_1, s_2 are sampled from the public source S, and correspond to hash functions $h_{s_1}, h_{s_2} \in \mathcal{H}$ mapping n-bit strings into m-bit strings.

4.3 Security Analysis

Here, we analyze the security of the immunizer described in Fig. 3. For input-constrained games, we obtain the following result whose proof appears in the full version. An analogous statement holds for input-unconstrained games, in the online watchdog model.

Theorem 1. *Let $\Pi = (\mathsf{P}, \mathsf{K}, \mathsf{R}, \mathsf{F}_1, \ldots, \mathsf{F}_N)$ be a deterministic cryptographic scheme, with $\mathcal{R} = \{0,1\}^m$, and consider any* input-constrained, single-instance *game $\mathbf{G} = (\mathsf{C}, \gamma)$ for Π. Then, for any $n, c^* > 4$, the immunizer $\Psi[\mathcal{G}, \mathsf{S}]$ of Fig. 3 is $(t_\mathsf{A}, t_\mathsf{W}, c^*, \epsilon^*)$-subversion-resistant with an* offline *watchdog, as long as $\mathcal{G} \stackrel{def}{=} \{g_s : \{0,1\}^n \to \{0,1\}^m\}_{s \in \{0,1\}^\ell}$ is a family of $(\frac{k}{n} \to \frac{m-d}{m}, t_{\text{cond}}, \epsilon_{\text{cond}})$-seed-dependent condensers and Π is either (t, ϵ)-secure w.r.t. game $\mathbf{G}$ (in case of unpredictability games) or (t, ϵ)-*square*-secure w.r.t. game $\mathbf{G}$ (in case of indistinguishability games), for parameters $t_{\text{cond}}, t, t_\mathsf{W} \approx t_\mathsf{A}$, and*

$$\epsilon \leq \begin{cases} \frac{c^*-1}{c^*} \cdot \frac{\epsilon^*}{2^{2d}} - \frac{2\epsilon_{\text{cond}}}{2^{2d}} & \text{if } \mathbf{G} \text{ is an unpredictability game} \\ \left(\frac{c^*-1}{c^*} \cdot \epsilon^* - 2\epsilon_{\text{cond}}\right)^2 \cdot \frac{1}{2^{2d}} & \text{if } \mathbf{G} \text{ is an indistinguishability game.} \end{cases}$$

Remark 3. Looking ahead, the reason for which Theorem 1 does not work for all deterministic primitives is that its proof crucially relies on the "overcoming weak expectations" framework. In particular, for single-instance indistinguishability games, this theorem requires square security, and it is well known that some primitives such as pseudorandom generators and pseudorandom functions do not have good square security [4,15].

Remark 4. The fact that our immunizer samples $2k$ times from the source R does *not* contradict the assumption that $\mathbf{G}$ is a single-instance game, as the latter condition only concerns the game $\mathbf{G}$ for the original primitive Π.

One can also show that the limitation of Remark 3 is inherent, in the sense that our immunizer is might be insecure for primitives that are not square friendly. Take, for instance, any PRG $\Pi = (\mathsf{R}, \mathsf{K}, \mathsf{PRG})$, where $\mathsf{K}(1^\lambda; r) = r$ outputs directly a seed sampled from the secret source R, and $\mathsf{PRG}(1^\lambda, r)$ stretches the seed to a pseudorandom output. Let $\Pi^* = (\mathsf{R}^*, \mathsf{K}^*, \mathsf{PRG}^*) = \Psi(\Pi)$ be the immunized version of Π. Now, consider the attacker $\mathsf{A}(s)$ that plays the subversion game by specifying the subversion $\widetilde{\Pi}$ where:

- $\widetilde{\mathsf{K}}$ and $\widetilde{\mathsf{PRG}}$ are unchanged (i.e., $\widetilde{\mathsf{K}} \equiv \mathsf{K}^*$, and $\widetilde{\mathsf{PRG}} \equiv \mathsf{PRG}^*$);
- $\widetilde{\mathsf{R}}$ embeds a key κ for a pseudorandom function PRF with one-bit output, and performs the following rejection-sampling procedure:
  - Sample a random r;
  - If $\mathsf{PRF}(1^\lambda, \kappa, y) = 1$, where $\mathsf{PRG}(h_s(r)) = y$, return r;
  - Else, sample a fresh r and start again.

Intuitively, the above subversion allows A to win the subversion game by simply checking whether $\mathsf{PRF}(1^\lambda, \kappa, y) = 1$, where y is the challenge. Moreover, this attack is undetectable as a watchdog not knowing the key κ has a negligible advantage in distinguishing $\widetilde{\mathsf{R}}$ from R^* (by the security of the pseudorandom function). Note that the above attack requires the adversary to choose the subversion depending on the seed.

Instantiating the Immunizer. When instantiating seed-dependent randomness condensers with state-of-the-art constructions [14,15], we obtain the following parameters.

Corollary 1. *For any cryptographic primitive Π that is either* $(\mathsf{poly}(\lambda),, \mathsf{negl}(\lambda))$*-secure (in case of unpredictability games) or* $(\mathsf{poly}(\lambda), \mathsf{negl}(\lambda))$-square*-secure (in case of indistinguishability games) w.r.t. an* input-constrained*,* single-instance *game* $\mathbf{G}$*, there exists an immunizer for Π that is* $(\mathsf{poly}(\lambda), \mathsf{poly}(\lambda), 5, \mathsf{negl}(\lambda))$*-subversion-resistant for the* pub*-model with an offline watchdog, with parameters $n, m, \ell \in \omega(\log(\lambda))$.*

Proof. By choosing $t, t_{\mathsf{A}}, t_{\mathsf{W}} \in \mathsf{poly}(\lambda)$, $\epsilon, \epsilon^* \in \mathsf{negl}(\lambda)$, $c^* = 5$, and setting $n \in \omega(\log(\lambda))$ in Theorem 1, we need a family of seed-dependent randomness condensers that achieves $t_{\mathrm{cond}} \in \mathsf{poly}(\lambda)$, $\epsilon_{\mathrm{cond}} \in \mathsf{negl}(\lambda)$, $k \in \omega(\log(\lambda))$, and entropy deficiency $d \in O(\log(\lambda))$.

Dodis, Ristenpart, and Vadhan [14] (see also [15]) have shown that any $(\mathsf{poly}(\lambda), \mathsf{poly}(\lambda)/2^m)$-collision-resistant family of hash functions directly yields such a family of condensers. The statement follows. □

5 Conclusions

We have shown how to immunize arbitrary *deterministic* cryptographic primitives against complete subversion, meaning that the adversary is allowed to tamper with all the underlying algorithms, and with the immunizer itself. In

the random oracle model, there is a simple immunizer that relies on a single secret, but tamperable, source of randomness [23,24]. In the standard model, instead, we need to assume an additional independent public, and in some case untamperable, random source.

Open problems include, e.g., finding better immunizers, both in terms of computational assumptions and/or the number of assumed trusted random sources. Also, exploring alternative approaches to achieve subversion security in the plain model for larger classes of cryptographic schemes (e.g., randomized ones), while still relying on $O(1)$ independent random sources, is an interesting direction for future research.

References

1. Ateniese, G., Magri, B., Venturi, D.: Subversion-resilient signature schemes. In: CCS, pp. 364–375 (2015)
2. Auerbach, B., Bellare, M., Kiltz, E.: Public-key encryption resistant to parameter subversion and its realization from efficiently-embeddable groups. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10769, pp. 348–377. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_12
3. Ball, J., Borger, J., Greenwald, G.: Revealed: how US and UK spy agencies defeat internet privacy and security. Guardian Weekly, September 2013
4. Barak, B., et al.: Leftover hash lemma, revisited. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 1–20. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_1
5. Bellare, M., Fuchsbauer, G., Scafuro, A.: NIZKs with an untrusted CRS: security in the face of parameter subversion. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 777–804. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_26
6. Bellare, M., Jaeger, J., Kane, D.: Mass-surveillance without the state: strongly undetectable algorithm-substitution attacks. In: CCS, pp. 1431–1440 (2015)
7. Bellare, M., Paterson, K.G., Rogaway, P.: Security of symmetric encryption against mass surveillance. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 1–19. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_1
8. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: CCS, pp. 62–73 (1993)
9. Checkoway, S., et al.: On the practical exploitability of dual EC in TLS implementations. In: USENIX Security Symposium, pp. 319–335 (2014)
10. Degabriele, J.P., Farshim, P., Poettering, B.: A more cautious approach to security against mass surveillance. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 579–598. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48116-5_28
11. Degabriele, J.P., Paterson, K.G., Schuldt, J.C.N., Woodage, J.: Backdoors in pseudorandom number generators: possibility and impossibility results. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 403–432. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_15

12. Dodis, Y., Ganesh, C., Golovnev, A., Juels, A., Ristenpart, T.: A formal treatment of backdoored pseudorandom generators. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 101–126. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_5
13. Dodis, Y., Mironov, I., Stephens-Davidowitz, N.: Message transmission with reverse firewalls - secure communication on corrupted machines. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 341–372. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_13
14. Dodis, Y., Ristenpart, T., Vadhan, S.P.: Randomness condensers for efficiently samplable, seed-dependent sources. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 618–635. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_35
15. Dodis, Y., Yu, Y.: Overcoming weak expectations. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 1–22. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_1
16. Fischlin, M., Janson, C., Mazaheri, S.: Backdoored hash functions: immunizing HMAC and HKDF. In: IEEE Computer Security Foundations Symposium, pp. 105–118 (2018)
17. Fischlin, M., Mazaheri, S.: Self-guarding cryptographic protocols against algorithm substitution attacks. In: IEEE Computer Security Foundations Symposium, pp. 76–90 (2018)
18. Greenwald, G.: No place to hide: Edward Snowden, the NSA, and the U.S. surveillance state. Metropolitan Books, May 2014
19. Hopper, N.J., von Ahn, L., Langford, J.: Provably secure steganography. IEEE Trans. Comput. **58**(5), 662–676 (2009)
20. Mironov, I., Stephens-Davidowitz, N.: Cryptographic reverse firewalls. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 657–686. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_22
21. Perlroth, N., Larson, J., Shane, S.: N.S.A. able to foil basic safeguards of privacy on web. The New York Times, September 2013
22. Rogaway, P.: The moral character of cryptographic work. IACR Cryptology ePrint Archive 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.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 34–64. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_2
24. Russell, A., Tang, Q., Yung, M., Zhou, H.: Generic semantic security against a kleptographic adversary. In: ACM CCS, pp. 907–922 (2017)
25. Russell, A., Tang, Q., Yung, M., Zhou, H.-S.: Correcting subverted random oracles. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 241–271. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_9
26. Simmons, G.J.: The Prisoners' problem and the subliminal channel. In: Chaum, D. (ed.) Advances in Cryptology, pp. 51–67. Springer, Boston (1984). https://doi.org/10.1007/978-1-4684-4730-9_5
27. Simmons, G.J.: The subliminal channel and digital signatures. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) EUROCRYPT 1984. LNCS, vol. 209, pp. 364–378. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39757-4_25
28. Trevisan, L., Vadhan, S.P.: Extracting randomness from samplable distributions. In: FOCS, pp. 32–42 (2000)

29. Young, A.L., Yung, M.: The dark side of "Black-Box" cryptography or: should we trust capstone? In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 89–103. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_8
30. Young, A., Yung, M.: Kleptography: using cryptography against cryptography. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 62–74. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_6