Handling Adaptive Compromise for Practical Encryption Schemes

Joseph Jaeger[1] and Nirvan Tyagi[2]

[1] Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle, USA
jsjaeger@cs.washington.edu
[2] Cornell Tech, New York City, USA
tyagi@cs.cornell.edu

Abstract. We provide a new definitional framework capturing the multi-user security of encryption schemes and pseudorandom functions in the face of adversaries that can adaptively compromise users' keys. We provide a sequence of results establishing the security of practical symmetric encryption schemes under adaptive compromise in the random oracle or ideal cipher model. The bulk of analysis complexity for adaptive compromise security is relegated to the analysis of lower-level primitives such as pseudorandom functions.
We apply our framework to give proofs of security for the BurnBox system for privacy in the face of border searches and the in-use searchable symmetric encryption scheme due to Cash et al. In both cases, prior analyses had bugs that our framework helps avoid.

Keywords: Symmetric cryptography, ideal models, adaptive security, searchable symmetric encryption

1 Introduction

A classic question in cryptography has been dealing with adversaries that adaptively compromise particular parties, thereby learning their secrets. Consider a setting where parties use keys $k_1, \ldots, k_n$ to encrypt messages $m_1, \ldots, m_n$ to derive ciphertexts $\mathsf{Enc}(k_1, m_1), \ldots, \mathsf{Enc}(k_n, m_n)$. An adversary obtains the ciphertexts and compromises a chosen subset of the parties to learn their keys. What can we say about the security of the messages encrypted by the keys that remain secret? Surprisingly, traditional approaches to formal security analysis, such as using encryption schemes that provide semantic security [19], fail to suffice for proving these messages' confidentiality. This problem was first discussed in the context of secure multiparty computation [10], and it arises in a variety of important cryptographic applications, as we explain below.

In this work, we introduce a new framework for formal analyses when security in the face of adaptive compromise is desired. Our approach provides a modular route towards analysis using idealized primitives (such as random oracles or ideal ciphers) for practical and in-use schemes. This modularity helps

us sidestep the pitfalls of prior ideal-model analyses that either invented new (less satisfying) ideal primitives, omitted proofs, or gave detailed but incorrect proofs. We exercise our framework across applications including searchable symmetric encryption (SSE), revocable cloud storage, and asymmetric password-authenticated key exchange (aPAKE). In particular, we provide full, correct proofs of security against adaptive adversaries for the Cash et al. [12] searchable symmetric encryption scheme that is used often in practice and the BurnBox system [33] for dealing with compelled-access searches. We show that our new definitions imply the notion of equivocable encryption introduced to prove security of the OPAQUE [24] asymmetric password-authenticated key exchange protocol. More broadly, our framework can be applied to a wide variety of constructions [1, 2, 9, 13, 17, 20, 21, 25–29, 34].

Current approaches to the "commitment problem". Our motivating applications have at their core an adaptive simulation-based model of security. Roughly speaking, they ask that no computationally bound adversary can distinguish between two worlds. In the first world, the adversary interacts with the scheme whose security is being measured. In the second world, the "ideal" world, the adversary's queries are instead handled by a simulator that must make do with only limited information which represents allowable "leakage" about the queries the adversary has made so far. The common unifying factor between varying applications we consider is that the adversary can make queries resulting in being given a ciphertexts encrypting messages of its choosing, then with future queries adaptively choose to expose the secret keys underlying some of the ciphertexts. The leakage given to the simulator will not include the messages encrypted unless a query has been made to expose the corresponding key.

Proving security in this model, however, does not work based on standard assumptions of the underlying encryption scheme. The problem is that the simulator must commit to ciphertexts, revealing them to the adversary, before knowing the messages associated to them. Hence the commitment problem. Several prior approaches for proving positive security results exist.

One natural approach attempts to build special non-committing encryption schemes [10] that can be proven (in the standard model) to allow opening some a priori fixed ciphertext to a message. But these schemes are not practical, as they require key material at least as long as the underlying message. Another unsatisfying approach considers only non-adaptive security in which an attacker specifies all of its queries at the beginning of the game. This is one of the two approaches that were simultaneously taken by Cash et al. [12]. Here the simulator is given the leakage for all of these queries at once and generates a transcript of all of its response. This is unsatisfying because more is lost when switching from adaptive to non-adaptive security than just avoiding the commitment problem. It is an easy exercise to construct encryption schemes which are secure when all queries to it must be chosen ahead of time but are not secure even against key-recovery attacks when an adversary may adaptively choose its queries.

The primary approach used to avoid this is to use idealized models, which we can again split into two versions. The first is to use an idealized model of

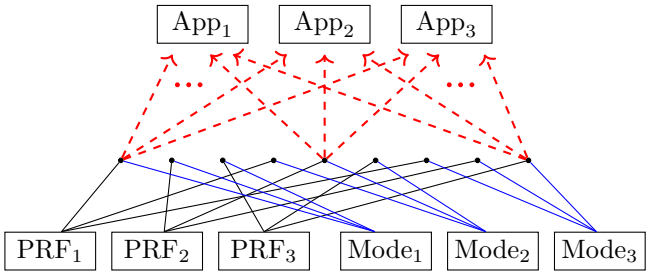

Fig. 1. Old state of affairs. Red dashed lines correspond to implications proved through programming in an ideal model proof. A different programming proof is needed to prove an application secure for each pair of PRF and symmetric encryption mode.

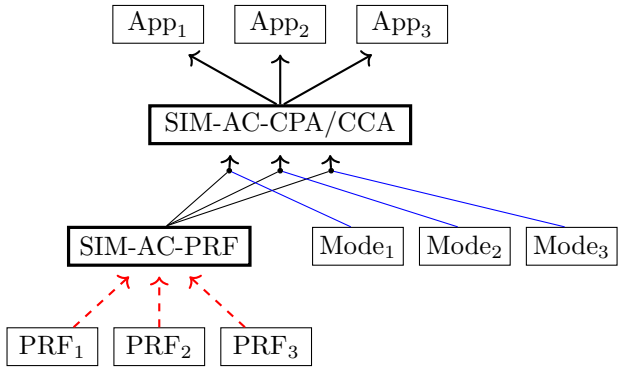

Fig. 2. New state of affairs. Red dashed lines correspond to implications proved through programming in an ideal model proof. New definitions are in bold boxes. Programming proofs are only needed to show each low level PRF construction meets SIM-AC-PRF.

encryption. Examples of this include indifferentiable authenticated encryption [3] (IAE) or the ideal encryption model (IEM) of Tyagi et al [33]. Security analyses in these models might not say much when one uses real encryption schemes, even when one is willing to use more established idealized models such as the ideal cipher model (ICM) or the random oracle model (ROM). One hope would be to use approaches such as indifferentiability [30] to modularly show that symmetric schemes sufficiently "behave like" ideal encryption, but this approach is unlikely to work for most encryption schemes used in practice [3].

The final approach, which is by far the most common in searchable symmetric encryption [1,2,9,12,13,17,20,21,25–29,34], is to fix a *particular encryption scheme* and prove security with respect to it in the ICM or ROM. Typically encryption schemes are built as modes of operations of an underlying pseudorandom function (PRF) and this function (or its constituent parts) is what is modeled as an ideal function. The downside of this is represented in Fig. 1. On the top, we have the applications one would like to prove secure, and on the

bottom, we have the different modes of operation and PRFs that one might use. Using this approach means that for each application, we have to provide a separate ideal model proof for each different choice of a mode of operation and a PRF (represented by dotted red arrows in Fig. 1). If there are A applications, P PRFs, and M modes of operation one might consider using, then this requires $A \cdot P \cdot M$ ideal model proofs in total, an unsatisfying state of affairs.

Moreover, the required ideal analysis can be tedious[3] and error-prone. This is presumably why only a few of the papers we found actually attempt to provide the full details of the ROM proof. We have identified bugs in all of the proofs that did provide the details. The lack of a full, valid proof among the fifteen papers we considered indicates the need for a more modular framework to capture this use of the random oracle. Our work provides such a framework, allowing the random oracle details to be abstracted away as a proof that only needs to be provided once. This framework provides definitions for use by other cryptographers that are simple to use, apply to practical encryption schemes, and allow showing adaptive security in well-studied models.

Examples of the "commitment problem". We proceed by discussing the example applications where we will apply our framework.

Revocable cloud storage and the compelled access setting. We start with the recently introduced compelled access setting (CAS) [33]. Here one wants encryption tools that provide privacy in the face of an authority searching digital devices, e.g., government searches of mobile phones or laptops at border crossings. To protect against compelled access searches, the BurnBox tool [33] uses what they call revocable encryption. At its core, this has the system encrypt a user's files $m_1, \ldots, m_n$ with independent keys $k_1, \ldots, k_n$. Ciphertexts are stored on (adversarially visible) cloud storage. Before a search occurs, the user instructs the application to delete the keys corresponding to files that the user wishes to hide from the authority, thereby revoking their own access to them. The other keys and file contents are disclosed to the authority.

The formal security definition introduced by Tyagi et al. captures confidentiality for revoked files even in the face of adversarial choice of which files to revoke, meaning they want security in the face of adaptive compromises. This very naturally results in the commitment problem because the simulator can be forced to provide ciphertexts for files, but only later learn the contents of these files at the time of key revelation. At which point, it is supposed to give keys which properly decrypt these ciphertexts.

To address the commitment problem they introduced the IEM. This models symmetric encryption as perfect: every encryption query is associated to a freshly chosen random string as ciphertext, and decryption is only allowed on previously returned ciphertexts. Analyses in the IEM can commit to ciphertexts (when the adversary doesn't know the key) and later open them to arbitrary messages. In their implementation, they used AES-GCM for encryption which cannot be

[3] Even more-so because SSE protocols often *also* run into the commitment problem with a PRF and need to model that using a random oracle as well.

thought of as indifferentiable from the IEM. Hence their proof can ultimately only provide heuristic evidence for the security of their implemenation.

Symmetric searchable encryption. Our second motivating setting is symmetric searchable encryption (SSE), which has similar issues as that discussed above for BurnBox, but with added complexity. SSE handles the following problem: a client wants to offload storage of a database of documents to an untrusted server while maintaining the ability to perform keyword searches on the database. The keyword searches should not reveal the contents of the documents to the server. To enable efficient solutions, we allow queries to leak some partial information about documents. Security is formalized via a simulation-based definition [15], in which a simulator given only the allowed leakage must trick an adversary into believing it is interacting with the actual SSE construction. An adaptive adversary can submit keyword searches as a function of prior returned results. Proving security here establishes that the scheme only leaks what is allowed and nothing more. While the leakage itself has been shown to be damaging in various contexts [11, 22], our focus here is on the formal analyses showing that leakage-abuse attacks are the best possible ones.

A common approach for SSE can be summarized at a high level as follows. The client generates a sequence of key pairs $(k_1, k'_1), \ldots, (k_n, k'_n)$ for keywords $w \in \{1, \ldots, n\}$ represented as integers for simplicity. The first key k_w in each pair is used to encrypt the identifiers of documents containing w. The latter key k'_w is used as a pseudorandom function (PRF) key to derive pseudorandom locations to store the encryption of the document identifiers. When the client later wants to search for documents containing w it sends the associated (k_w, k'_w) keys to the server. The server then uses k'_w to re-derive the pseudorandom locations of the ciphertexts and uses k_w to decrypt them.

To prove adaptive security, the simulator for such a protocol runs into the commitment problem because it must commit to ciphertexts of the document identifiers before knowing what the identifiers are. Perhaps less obviously, a simulator also runs into a commitment issues with the PRF. To ensure security the simulated locations of ciphertexts must be random, but then when responding to a search query the simulator is on the hook to find a key for the PRF that "explains" the simulated locations. Papers on SSE typically address these issue by modeling the PRF as a random oracle and fixing a specific construction of an encryption scheme based on a random oracle. As noted earlier, this has resulted in a need for many tedious and error-prone proofs.

Asymmetric password-authenticated key exchange and equivocable encryption. In independent and concurrent work, Jarecki et al. updated the ePrint version of [24] to introduce the notion of equivocable encryption and use it to prove security of their asymmetric password-authenticated key exchange protocol OPAQUE. The definition of equivocable encryption is essentially a weakened version of our confidentiality definition, considering only single-user security and allowing only a single encryption query; whereas we consider multi-user security and arbitrarily many adaptively chosen queries. Since their definition is implied

by ours, our results will make rigorous their claim that "common encryption modes are equivocable under some idealized assumption".

A new approach. We introduce a new framework for analyzing security in adaptive compromise scenarios. Our framework has a simple, but powerful recipe: augment traditional simulation-based, property-based security definitions to empower adversaries with the ability to perform adaptive compromise of secret keys. For symmetric encryption, for example, we convert the standard simulation-based, multi-user indistinguishability under chosen plaintext attack (mu-IND-CPA) [4] to a game that includes the same adversarial powers, but adds an additional oracle for adaptively compromising individual user secret keys. Critical to our approach is (1) the use of simulators, which allows handling corruptions gracefully, and (2) incorporating handling of idealized models (e.g., the ROM or ICM). The latter is requisite for analyzing practical constructions.

We offer new definitions for multi-user CPA and CCA security of symmetric encryption, called SIM-AC-CPA (simulation-based security under adaptive corruption, chosen plaintext attack) and SIM-AC-CCA (chosen ciphertext attack). By restricting the classes of allowed simulators we can obtain stronger definitions (e.g., SIM-AC-$ which requires that ciphertexts look like random strings).

Symmetric encryption under adaptive compromise. We then begin exercising our framework by first answering the question: Are practical, in-use symmetric encryption schemes secure in the face of adaptive compromises? We give positive results here, in idealized models. Taking an encrypt-then-MAC scheme such as AES in counter mode combined with HMAC [5] as an example, we could directly show SIM-AC-CCA security while modeling AES as an ideal cipher and HMAC as a random oracle (c.f., [16]). But this would lead to a rather complex proof, and we'd have to do similarly complex proofs for other encryption schemes.

Instead, we provide simple, modular proofs by lifting the underlying assumptions made about primitives (AES and HMAC) to hold in adaptive compromise scenarios. Specifically, we introduce a new security notion for pseudorandom functions under adaptive compromise attacks (SIM-AC-PRF). This adapts the standard multi-user PRF notion to also give adversaries the ability to adaptively compromise particular keys. Then we prove that AES and HMAC each achieve this notion in the ICM and ROM, respectively. The benefit is that these proofs encapsulate the complexity of ideal model programming proofs in the simpler context of SIM-AC-PRF (as opposed to SIM-AC-CCA).

The workflow when using our framework is represented by Fig. 2. Here PRFs are individually shown to achieve SIM-AC-PRF security in an ideal model. Then modes of operation are proven secure under the assumption that they use a SIM-AC-PRF secure PRF. Then each application is proven secure under the appropriate assumption of the encryption scheme used. This decreases the total number of proofs required to $A + P + M$, significantly fewer than the $A \cdot P \cdot M$ required previously. Moreover, the complicated ideal model programming analysis (represented by red dashed arrows) is restricted to only appearing in the the simplest of these proofs (analyzing of PRFs); it can then simply be "passed along" to the higher level proofs.

We can then show that for most CPA modes of operation (e.g., CBC mode or CTR mode), one can prove SIM-AC-CPA security assuming the underlying block cipher is SIM-AC-PRF. The core requirement is that the mode of operation enjoys a property that we call extraction security. This is a technical condition capturing the key security properties needed to prove that a mode of operation is SIM-AC-CPA assuming the underlying block cipher is SIM-AC-PRF. Moreover, we show that most existing (standard) proofs of IND-CPA security show, implicitly, the extraction security of the mode. Thus, we can easily establish adaptive compromise proofs given existing (standard) ones.

The above addresses only confidentiality. Luckily, integrity is inherited essentially for free from existing analysis. We generically show that SIM-AC-CPA security combined with the standard notion of ciphertext integrity implies SIM-AC-CCA security. Thus, one can prove encrypt-then-MAC is SIM-AC-CCA secure assuming the SIM-AC-CPA security of the encryption and the standard unforgeability security of the MAC. This is an easy adaptation of the standard proof [8] of encrypt-then-MAC.

Applying the framework to high-level protocols. Equipped with our new SIM-AC-CCA and SIM-AC-PRF security notions, we can return to our motivating task: providing positive security analyses of BurnBox and the Cash et al. SSE scheme.

We give a proof of BurnBox's CAS security assuming the SIM-AC-CPA security of the underlying symmetric encryption scheme. Our proof is significantly simpler than the original analysis, avoiding specifically the nuanced programming logic that led to the bug in the original analysis. For the Cash et al. scheme we apply our SIM-AC-PRF definition and a key-private version of our SIM-AC-CPA definition. Their adaptive security claim was accompanied only by a brief proof sketch which fails to identify an important detail that need to be considered in the ROM analysis (see the full version of this paper [23]). Our proof handles this detail cleanly while being ultimately of comparable complexity to their non-adaptive security proof.

Unfortunately, these settings and constructions are inherently complicated. So even with the simplification provided by our analysis techniques there is not space to fit their analysis in the body of our paper; it has instead been relegated to the full version of this work. We choose this organization because our main contribution is the definition abstraction which we believe will be of use for future work, rather than the particular applications we chose to exhibit its use.

Treatment of symmetric encryption. In this work, we focus on randomized encryption, over more modern nonce-based variants because this was the form of encryption used by the applications we identified. In the full version of this paper [23], we extend our definitions to nonce-based encryption. The techniques we introduce for analyzing randomized symmetric encryption schemes should extend to nonce-based encryption schemes.

Related works. A related line of work is that of selective-opening attacks [7] which studies the security of asymmetric encryption schemes against compromises in a multi-sender setting (where coins underlying encryption may be compromised) or multi-receiver setting (where secret decryption keys may be com-

promised). Selective-opening definitions are typically formulated to aim for standard model (or non-programmable random oracle model) achievability and hence do not suffice for the applications we consider in this work.

The full version of this paper is available on ePrint [23].

2 Preliminaries

A list T of length $n \in \mathbb{N}$ specifies an ordered sequence of elements $T[1]$, $T[2]$, $\ldots$, $T[n]$. The operation $T.\mathsf{add}(x)$ appends x to this list by setting $T[n+1] \leftarrow x$. This making T a list of length $n+1$. We let $|T|$ denote the length of T. The operation $x \leftarrow T.\mathsf{dq}()$ sets x equal to the last element of T and removes this element from T. In pseudocode lists are assumed to be initialized empty (i.e. have length 0). An empty list or table is denoted by $[\cdot]$. We sometimes use set notation with a list, e.g. $x \in T$ is true if $x = T[i]$ for any $1 \leq i \leq |T|$.

Let S and S' be two sets with $|S| \leq |S'|$. Then $\mathsf{Inj}(S, S')$ is the set of all injections from S to S'. We will sometimes abuse terminology and refer to functions with co-domain $\{\, S : S \subseteq \{0,1\}^* \,\}$ as sets. For $n \in \mathbb{N}$ we define $[n] = \{1, \ldots, n\}$.

The notation $y \leftarrow\!\!\$\ A(x_1, x_2, \cdots : \sigma)$ denotes the (randomized) execution of A with state σ. Changes that A makes to its input variable σ are maintained after A's execution. For given $x_1, x_2, \ldots$ and σ we let $[A(x_1, x_2, \cdots : \sigma)]$ denote the set of possible outputs of A given these inputs.

We define security notions using pseudocode-based games. The pseudocode "Require bool" is shorthand for "If not bool then return $\perp$". We will sometimes use infinite loops defining variable x_u for all $u \in \{0,1\}^*$. Such code is executed lazily; the code is initially skipped, then later if a variable x_u would be used, the necessary code to define it is first run. The pseudocode "$\exists x \in X$ s.t. $p(x)$" for some predicate p evaluates to the boolean value $\bigvee_{x \in X} p(x)$. If this is true, the variable x is set equal to the lexicographically first $x \in X$ for which $p(x)$ is true.

We use an asymptotic formalism. The security parameter is denoted λ. Our work is generally written in a way to allow concrete security bounds to be extracted easily. In security proofs we typically explicitly state how we will bound the advantage of an adversary by the advantages of reduction adversaries we build (and possibly other terms). Reduction adversaries and simulators are explicitly given in code (from which concrete statements about their efficiency can be obtained by observation).

Let $f : \mathbb{N} \to \mathbb{N}$. We say f is negligible if for all polynomials p there exists a $\lambda_p \in \mathbb{N}$ such that $f(\lambda) \leq 1/p(\lambda)$ for all $\lambda \geq \lambda_p$. We say f is super-polynomial if $1/f$ is negligible. We say f is super-logarithmic if 2^f is super-polynomial.

Ideal primitives. We will make liberal use of ideal primitives such as random oracles or ideal ciphers. An ideal primitive P specifies algorithms $\mathsf{P.Init}$ and $\mathsf{P.Prim}$. The initialization algorithm has syntax $\sigma_{\mathsf{P}} \leftarrow\!\!\$\ \mathsf{P.Init}(1^\lambda)$. The stateful evaluation algorithm has syntax $y \leftarrow\!\!\$\ \mathsf{P.Prim}(1^\lambda, x : \sigma_{\mathsf{P}})$. We sometimes us A^{P} as shorthand for giving algorithm A oracle access to $\mathsf{P.Prim}(1^\lambda, \cdot : \sigma_{\mathsf{P}})$. Adversaries are often given access to P via an oracle PRIM.

Ideal primitives should be *stateless*. By this we mean that after σ_{P} is output by $\mathsf{P}.\mathsf{Init}$, it is never modified by $\mathsf{P}.\mathsf{Prim}$ (so we could have used the syntax $y \leftarrow\!\!\$\ \mathsf{P}.\mathsf{Prim}(1^\lambda, x, \sigma_{\mathsf{P}})$). However, when written this way, ideal primitives are typically inefficient, e.g., for the random oracle model σ_{P} would store a huge random table. Our security results will necessitate that P be efficiently instantiated so we have adopted the stateful syntax to allow ideal primitives to be written in their efficient "lazily sampled" form. Despite this notational convenience, we will assume that any ideal primitive we reference is *essentially stateless*. By this, we mean that it could have been equivalently written to be stateless (if inefficient).[4]

The standard model is captured by the primitive P_{sm} for which $\mathsf{P}_{\mathsf{sm}}.\mathsf{Init}(1^\lambda)$ and $\mathsf{P}_{\mathsf{sm}}.\mathsf{Prim}(1^\lambda, x : \sigma_{\mathsf{P}})$ always returns the empty string ε.

We define a random oracle that takes arbitrary input and produce variable length outputs. It is captured by the primitive $\mathsf{P}_{\mathsf{rom}}$ defined as follows.

$$\begin{array}{l|l}
\underline{\mathsf{P}_{\mathsf{rom}}.\mathsf{Init}(1^\lambda)} & \underline{\mathsf{P}_{\mathsf{rom}}.\mathsf{Prim}(1^\lambda, x : T)} \\
\text{Return } [\cdot] & (x, l) \leftarrow x \\
 & \text{If } T[x, l] = \bot \text{ then } T[x, l] \leftarrow\!\!\$\ \{0,1\}^l \\
 & \text{Return } T[x, l]
\end{array}$$

The ideal-cipher model is parameterized by a block-length $n : \mathbb{N} \to \mathbb{N}$ and captured by $\mathsf{P}^n_{\mathsf{icm}}$ defined as follows.[5]

$$\begin{array}{l|l}
\underline{\mathsf{P}^{\mathsf{n}}_{\mathsf{icm}}.\mathsf{Init}(1^\lambda)} & \underline{\mathsf{P}^{\mathsf{n}}_{\mathsf{icm}}.\mathsf{Prim}(1^\lambda, x : (E, D))} \\
\text{Return } ([\cdot], [\cdot]) & (\mathrm{op}, K, y) \leftarrow x \\
 & \text{If op} = + \text{ then} \\
 & \quad \text{If } E[K, y] = \bot \text{ then} \\
 & \quad\quad z \leftarrow\!\!\$\ \{0,1\}^{n(\lambda)} \setminus \{\ E[K, a]\ :\ a \in \{0,1\}^{n(\lambda)}\ \} \\
 & \quad\quad E[K, y] \leftarrow z\ ;\ D[K, z] \leftarrow y \\
 & \quad \text{Return } E[K, y] \\
 & \text{Else} \\
 & \quad \text{If } D[K, y] = \bot \text{ then} \\
 & \quad\quad z \leftarrow\!\!\$\ \{0,1\}^{n(\lambda)} \setminus \{\ D[K, a]\ :\ a \in \{0,1\}^{n(\lambda)}\ \} \\
 & \quad\quad D[K, y] \leftarrow z\ ;\ E[K, z] \leftarrow y \\
 & \quad \text{Return } D[K, y]
\end{array}$$

It stores tables E and D which it uses to lazily sample a random permutation for each K, with $E[K, \cdot]$ representing the forward evaluation and $D[K, \cdot]$ its inverse. It parses its input as a tuple (op, K, y) where $\mathrm{op} \in \{+, -\}$ specifies the direction of evaluation and $K \in \{0,1\}^*$ and $y \in \{0,1\}^{n(\lambda)}$ specify the input.

Sometimes we construct a cryptographic primitive from multiple underlying cryptographic primitives which expect different ideal primitives. To capture this it will be useful to have a notion of combining ideal primitives. Let P' and P'' be ideal primitives. We define their cross product $\mathsf{P} = \mathsf{P}' \times \mathsf{P}''$ as follows.

[4] Without this restrictions an ideal primitive could behave in undesirable, contrived ways (e.g., on some special input outputting all prior inputs it has received).

[5] We will implicitly assume $n(\lambda)$ can be computed in time polynomial in λ. We make similar implicit assumptions for other functions that parameterize constructions of cryptographic primitives.

$\underline{\mathsf{P}.\mathsf{Init}(1^\lambda)}$	$\underline{\mathsf{P}.\mathsf{Prim}(1^\lambda, x : \sigma_\mathsf{P})}$
$\sigma'_\mathsf{P} \leftarrow\!\!\$\ \mathsf{P}'.\mathsf{Init}(1^\lambda)$	$(\sigma'_\mathsf{P}, \sigma''_\mathsf{P}) \leftarrow \sigma_\mathsf{P}$
$\sigma''_\mathsf{P} \leftarrow\!\!\$\ \mathsf{P}''.\mathsf{Init}(1^\lambda)$	$(d, x) \leftarrow x$
Return $(\sigma'_\mathsf{P}, \sigma''_\mathsf{P})$	If $d = 1$ then $y \leftarrow\!\!\$\ \mathsf{P}'.\mathsf{Prim}(1^\lambda, x : \sigma'_\mathsf{P})$
	Else $y \leftarrow\!\!\$\ \mathsf{P}''.\mathsf{Prim}(1^\lambda, x : \sigma''_\mathsf{P})$
	$\sigma_\mathsf{P} \leftarrow (\sigma'_\mathsf{P}, \sigma''_\mathsf{P})$
	Return y

By our earlier convention $A^{\mathsf{P}' \times \mathsf{P}''}$ is shorthand for giving algorithm A oracle access to $\mathsf{P}.\mathsf{Prim}(1^\lambda, \cdot : \sigma_\mathsf{P})$. In A's code, $B^{\mathsf{P}'}$ denotes giving B oracle access to $\mathsf{P}.\mathsf{Prim}(1^\lambda, (1, \cdot) : \sigma_\mathsf{P})$ and $B^{\mathsf{P}''}$ to denote giving B oracle access to $\mathsf{P}.\mathsf{Prim}(1^\lambda, (2, \cdot) : \sigma_\mathsf{P})$.

2.1 Standard Cryptographic Definitions

We recall standard cryptographic syntax and security notions.

Symmetric encryption syntax. A symmetric encryption scheme SE specifies algorithms $\mathsf{SE}.\mathsf{Kg}$, $\mathsf{SE}.\mathsf{Enc}$, and $\mathsf{SE}.\mathsf{Dec}$ as well as sets $\mathsf{SE}.\mathsf{M}$, $\mathsf{SE}.\mathsf{Out}$, and $\mathsf{SE}.\mathsf{K}$ representing the message, ciphertext, and key space respectively. The key generation algorithm has syntax $K \leftarrow\!\!\$\ \mathsf{SE}.\mathsf{Kg}(1^\lambda)$. The encryption algorithm has syntax $c \leftarrow\!\!\$\ \mathsf{SE}.\mathsf{Enc}^\mathsf{P}(1^\lambda, K, m)$, where $c \in \mathsf{SE}.\mathsf{Out}(\lambda, |m|)$ is required. The deterministic decryption algorithm and has syntax $m \leftarrow \mathsf{SE}.\mathsf{Dec}^\mathsf{P}(1^\lambda, K, c)$. Rejection of c is represented by returning $m = \bot$. Informally, correctness requires that encryptions of messages in $\mathsf{SE}.\mathsf{M}(\lambda)$ decrypt properly. We assume the boolean $(m \in \mathsf{SE}.\mathsf{M}(\lambda))$ can be efficiently computed.

Integrity of ciphertexts. Integrity of ciphertext security is defined by the game $\mathrm{G}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathcal{A}_{\mathsf{ctxt}}}$ shown in Fig. 3. In the game, the attacker interacts with one of two "worlds" (determined by the bit b) via its oracles Enc, Prim, Exp, and Dec. The attacker's goal is to determine which world it is interacting with.

$\underline{\text{Game } \mathrm{G}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathsf{P},\mathcal{A}_{\mathsf{ctxt}}}(\lambda)}$	$\underline{\textsc{Enc}(u, m)}$	$\underline{\textsc{Dec}(u, c)}$
For $u \in \{0,1\}^*$ do	Require $m \in \mathsf{SE}.\mathsf{M}(\lambda)$	Require $u \notin X$
$\quad K_u \leftarrow\!\!\$\ \mathsf{SE}.\mathsf{Kg}(1^\lambda)$	$c \leftarrow\!\!\$\ \mathsf{SE}.\mathsf{Enc}^\mathsf{P}(1^\lambda, K_u, m)$	Require $c \notin C_u$
$\sigma_\mathsf{P} \leftarrow\!\!\$\ \mathsf{P}.\mathsf{Init}(1^\lambda)$	$C_u.\mathsf{add}(c)$	$m_1 \leftarrow \mathsf{SE}.\mathsf{Dec}^\mathsf{P}(1^\lambda, K_u, c)$
$b \leftarrow\!\!\$\ \{0,1\}$	Return c	$m_0 \leftarrow \bot$
$b' \leftarrow\!\!\$\ \mathcal{A}^{\textsc{Enc},\textsc{Dec},\textsc{Exp},\textsc{Prim}}_{\mathsf{ctxt}}(1^\lambda)$	$\underline{\textsc{Exp}(u)}$	Return m_b
Return $(b = b')$	$X.\mathsf{add}(u)$	
$\underline{\textsc{Prim}(x)}$	Return K_u	
$y \leftarrow\!\!\$\ \mathsf{P}.\mathsf{Prim}(1^\lambda, x : \sigma_\mathsf{P})$		
Return y		

Fig. 3. Game defining multi-user CTXT security of SE in the face of exposures.

The PRIM oracle gives the attacker access to the ideal primitive P. The encryption oracle ENC takes as input a user u and message m, then returns the encryption of that message using the key of that user, K_u. Recall that by our convention each K_u is not sampled until needed. The exposure oracle EXP takes in u and then returns K_u to the attacker. The decryption oracle DEC is the only oracle whose behavior depends on the bit b. It takes as input a user u and ciphertext c. When $b = 1$, it will return the decryption of c using K_u while when $b = 0$ it will always return $\perp$. Thus, the goal of the attacker it to forge a ciphertext which will decrypt to a non-$\perp$ value.

To prevent trivial attacker, we disallow querying a ciphertext to $\textsc{Dec}(u, \cdot)$ if it came from $\textsc{Enc}(u, \cdot)$ or if u was already exposed. This is captured by the "Require" statements in DEC using lists C_u and X (which store the ciphertexts returned by $\textsc{Enc}(u, \cdot)$ and the users that have been exposed, respectively).

We define the advantage function $\mathsf{Adv}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathsf{P},\mathcal{A}_{\mathsf{ctxt}}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathsf{P},\mathcal{A}_{\mathsf{ctxt}}}(\lambda)] - 1$. We say SE is INT-CTXT secure with P if for all PPT $\mathcal{A}_{\mathsf{ctxt}}$, the advantage $\mathsf{Adv}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathsf{P},\mathcal{A}_{\mathsf{ctxt}}}(\cdot)$ is negligible. INT-CTXT security is typically defined to only consider a single user and no exposures. Using a hybrid argument one can show that our definition of INT-CTXT security is implied by the more standard definition.

Function family. A family of functions F specifies algorithms $\mathsf{F.Kg}$ and $\mathsf{F.Ev}$ together with sets $\mathsf{F.Inp}$ and $\mathsf{F.Out}$. The key generation algorithm has syntax $K \leftarrow\!\!\$\ \mathsf{F.Kg}^{\mathsf{P}}(1^\lambda)$. The evaluation algorithm is deterministic and has the syntax $y \leftarrow \mathsf{F.Ev}(1^\lambda, K, x)$. It is required that for all $\lambda \in \mathbb{N}$ and $K \in [\mathsf{F.Kg}(1^\lambda)]$ that $\mathsf{F.Ev}(1^\lambda, K, x) \in \mathsf{F.Out}(\lambda)$ whenever $x \in \mathsf{F.Inp}(\lambda)$. It is assumed that random elements of $\mathsf{F.Out}(\lambda)$ can be efficiently sampled.

Game $\mathrm{G}^{\mathsf{ow}}_{\mathsf{F},\mathsf{P},\mathcal{A}}(\lambda)$	$\textsc{Prim}(x)$
$K \leftarrow\!\!\$\ \mathsf{F.Kg}(1^\lambda)\ ;\ \sigma_{\mathsf{P}} \leftarrow\!\!\$\ \mathsf{P.Init}(1^\lambda)$	$y \leftarrow\!\!\$\ \mathsf{P.Prim}(1^\lambda, x : \sigma_{\mathsf{P}})$
$x \leftarrow\!\!\$\ \mathsf{F.Inp}(\lambda)\ ;\ y \leftarrow \mathsf{F.Ev}^{\mathsf{P}}(\lambda, K, x)$	Return y
$x' \leftarrow\!\!\$\ \mathcal{A}^{\textsc{Prim}}(1^\lambda, K, y)$	
Return $(\mathsf{F.Ev}^{\mathsf{P}}(1^\lambda, K, x') = y)$	

Fig. 4. Game defining one-wayness of F.

One-wayness. The one-wayness of a family of functions F is given by the game G^{ow} shown in Fig. 4. The adversary is given a key K to F and the image y of a random point x in the domain. Its goal is to find a point with the same image. We define the advantage function $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathsf{P},\mathcal{A}}(\lambda) = \Pr[\mathrm{G}^{\mathsf{ow}}_{\mathsf{F},\mathsf{P},\mathcal{A}}(\lambda)]$ and say F is OW secure with P if $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathsf{P},\mathcal{A}}(\cdot)$ is negligible for all PPT $\mathcal{A}$.

Security definitions. In the body of this paper we sometimes informally reference other security notions for symmetric encryption schemes (IND-CPA, IND-CCA, IND-KP, IND-$) and function families (PRF, UF-CMA). These definitions are recalled in the full version of this paper [23].

3 New Security Definitions for Symmetric Primitives

In this section we provide our definitions for the security of symmetric cryptographic primitives (namely randomized encryption and pseudorandom functions) against attackers able to adaptively compromise users' keys.

3.1 Randomized Symmetric Encryption

We describe our security definitions for randomized symmetric encryption. We refer to them as SIM-AC-CPA and SIM-AC-CCA security. The definition of SIM-AC-CPA (resp. SIM-AC-CCA) security is a generalization of IND-CPA (IND-CCA) security to a multi-user setting in which some users' keys may be compromised by an attacker.

Consider game $\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}cpa}}$ shown in Fig. 5. It is parameterized by a symmetric encryption scheme SE, simulator S, ideal primitive P, and attacker $\mathcal{A}_{\mathsf{cpa}}$. The attacker interacts with one of two "worlds" via its oracles Enc, Exp, and Prim. The attacker's goal is to determine which world it is interacting with.

In the real world ($b = 1$) the encryption oracle Enc takes (u, m) as input and returns an encryption of m using u's key K_u. Oracle Prim returns the output of the ideal primitive on input x. Oracle Exp returns u's key K_u to the attacker.

In the ideal world ($b = 0$), the return values of each of these oracles are instead chosen by a simulator S. In Prim it is given the input provided to the oracle. In Enc it is given the name of the current user u and some leakage ℓ about the message m. If u has not yet been exposed ($u \notin X$) this leakage is just the length of the message. Otherwise the leakage is the message itself. The inputs and outputs of this oracle for a user u are stored in the lists M_u and C_u so they can be leaked to the simulator when $\textsc{Exp}(u)$ is called.

We define $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\lambda)] - 1$. We say SE is SIM-AC-CPA secure with P if for all PPT $\mathcal{A}_{\mathsf{cpa}}$ there exists a PPT S such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\cdot)$ is negligible. Intuitively, this definition captures that ciphertexts reveal nothing about the messages encrypted other than their length unless the encryption key is known to the attacker. In the full version of this paper [23], we show that SIM-AC-CPA security is impossible in the standard model. The proof is a simple application of the ideas of Nielsen [31].

SIM-AC-CCA security extends SIM-AC-CPA security by giving $\mathcal{A}_{\mathsf{cca}}$ access to a decryption oracle which takes as input (u, c). In the real world, it returns the decryption of c using K_u. In the ideal world, the simulator simulates this. To prevent trivial attacks, the attacker is disallowed from querying (u, c) if c was returned from an earlier query $\textsc{Enc}(u, m)$. We define $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cca}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cca}}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}cca}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cca}}}(\lambda)] - 1$. We say SE is SIM-AC-CCA secure with P if for all PPT $\mathcal{A}_{\mathsf{cca}}$ there exists a PPT S such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cca}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cca}}}(\cdot)$ is negligible.

Simplifications. It will be useful to keep in mind simplifications we can make to restrict the behavior of the adversary or simulator without loss of generally. They are applicable to all SIM-AC-style definitions we provide in this paper.

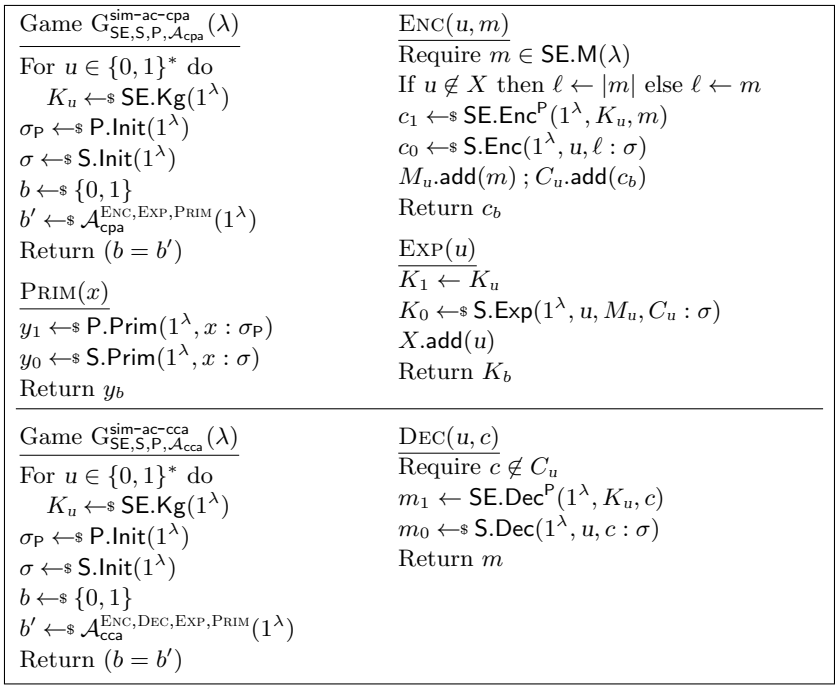

Fig. 5. Games defining SIM-AC-CPA and SIM-AC-CCA security of SE.

- If an oracle is deterministic in the real world, then we can assume that the adversary never repeats a query to this oracle or that the simulator always provides the same output to repeated queries.
- We can assume the adversary never makes a query to a user it has already exposed or that for such queries the simulator just runs the code of the real world (replacing calls to P with calls to $\mathsf{S.Prim}$).
- We can assume the adversary always queries with $u \in [u_\lambda]$ for some polynomial $u_{(\cdot)}$ or that the simulator is agnostic to the particular strings used to reference users.
- We can assume that adversaries never make queries that fail "Require" statements. (All requirements of oracles we provide will be efficiently computable given the transcripts of queries the adversary has made.)

Proving these are slightly more subtle to establish than analogous simplifications would be in non-simulation-based games because of the order that algorithms are quantified in our security definitions. They all follow the same pattern though, so we sketch the second of these.

Suppose SE is SIM-AC-CPA secure for all adversaries that never make a call $\textsc{Enc}(u, m)$ after having made a call $\textsc{Exp}(u)$, then we claim SE is SIM-AC-CPA secure. Let $\mathcal{A}$ be an arbitrary adversary. Then we build a wrapper adversary $\mathcal{A}'$ that simply forwards all of $\mathcal{A}$'s queries except for encryption queries made

for a user that has already been exposed. In these cases $\mathcal{B}$ responds with the output of $\mathsf{SE.Enc}^{\textsc{Prim}(\cdot)}(1^\lambda, K_u, m)$ (or $\perp$ if $m \notin \mathsf{SE.M}(\lambda)$), where K_u is the key last returned from $\textsc{Exp}(u)$. Let S' be a simulator for $\mathcal{A}'$. Then we construct S for $\mathcal{A}$ which responds exactly as S' would except in response to encryption queries made for a user that has already been exposed. In these cases S' responds with the output of $\mathsf{SE.Enc}^{\mathsf{S}'.\mathsf{Prim}(1^\lambda,\cdot,\cdot:\sigma)}(1^\lambda, K_u, m)$, where K_u is the key it last returned for $\textsc{Exp}(u)$. It is clear that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE,S,P},\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S}',\mathsf{P},\mathcal{A}'}(\lambda)$ because the view of $\mathcal{A}$ is identical in the corresponding games.

Stronger security notions. It is common in the study of symmetric encryption primitives to study stronger security definitions than IND-CPA security. Most schemes instead aim directly for their output to be indistinguishable from random bits (IND-$). This implies IND-CPA security and additional nice properties such as forms of key-privacy.

We can capture such notions by placing restrictions on the behavior of the simulator. Let S be a simulator (for which we think of $\mathsf{S.Enc}$ as being undefined) which additionally defines algorithms $\mathsf{S.Enc}_1$ and $\mathsf{S.Enc}_2$ as well as set $\mathsf{S.Out}$. Then we define simulators $\mathsf{S}_\mathsf{k}[\mathsf{S}]$ and $\mathsf{S}_\$[\mathsf{S}]$ to be identical to S except for the following encryption simulation algorithms.

$\underline{\mathsf{S}_\mathsf{k}[\mathsf{S}].\mathsf{Enc}(1^\lambda, u, \ell : \sigma)}$	$\underline{\mathsf{S}_\$[\mathsf{S}].\mathsf{Enc}(1^\lambda, u, \ell : \sigma)}$
If $\ell \in \mathbb{N}$ then $c \leftarrow\!\!\$\ \mathsf{S.Enc}_1(1^\lambda, \ell : \sigma)$	If $\ell \in \mathbb{N}$ then $c \leftarrow\!\!\$\ \mathsf{S.Out}(\lambda, \ell)$
Else $c \leftarrow\!\!\$\ \mathsf{S.Enc}_2(1^\lambda, u, \ell : \sigma)$	Else $c \leftarrow\!\!\$\ \mathsf{S.Enc}_2(1^\lambda, u, \ell : \sigma)$
Return c	Return c

Checking $\ell \in \mathbb{N}$ acts as a convenient way of verifying if the user being queried has been exposed yet. Because $\mathsf{S.Enc}_1(1^\lambda, \ell : \sigma)$ is not given u in S_k, the output of S_k is distributed identically for any unexposed users. The class of key-anonymous simulators $\mathcal{S}_k$ is the set of all $\mathsf{S}_\mathsf{k}[\mathsf{S}]$ for some S. Similarly, $\mathsf{S}_\$$ always outputs a random bitstring as the ciphertext for any unexposed user. The class of random-ciphertext simulators $\mathcal{S}_\$$ is the set of all $\mathsf{S}_\$[\mathsf{S}]$ for some S. Note that $\mathcal{S}_\$ \subset \mathcal{S}_k$.

We say SE is SIM-AC-KP secure with P if for all PPT $\mathcal{A}_{\mathsf{cpa}}$ there exists a PPT $\mathsf{S} \in \mathcal{S}_k$ such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE,S,P},\mathcal{A}_{\mathsf{cpa}}}(\cdot)$ is negligible. We say that SE is SIM-AC-$ secure with P if for all PPT $\mathcal{A}_{\mathsf{cpa}}$ there exists a PPT $\mathsf{S} \in \mathcal{S}_\$$ such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE,S,P},\mathcal{A}_{\mathsf{cpa}}}(\cdot)$ is negligible. It is straightforward to see that SIM-AC-$ security implies SIM-AC-KP security which in turn implies SIM-AC-CPA security. Standard counter-examples will show that these implications do not hold in the other direction.

It is sometimes useful to define security in an all-in-one style, introduced by Rogaway and Shrimpton [32], which simultaneously requires IND-$ security and INT-CTXT security. In our framework we can define $\mathcal{S}_\perp$ as the class of IND-CCA simulators which always return $\perp$ for decryption queries to unexposed users. Then we say SE is SIM-AC-AE secure with P if for all PPT $\mathcal{A}_{\mathsf{cca}}$ there exists a PPT $\mathsf{S} \in \mathcal{S}_\$ \cap \mathcal{S}_\perp$ such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cca}}_{\mathsf{SE,S,P},\mathcal{A}_{\mathsf{cca}}}(\cdot)$ is negligible.

3.2 Pseudorandom Functions

Typically a symmetric encryption scheme will use a PRF as one of their basic building blocks. For modularity, it will be useful to provide a simulation-based security definition for PRFs in the face of active compromises. In Section 6, we show our PRF definition can be applied to construct a SIM-AC secure symmetric encryption scheme. Additionally, in the full version of this paper [23], we show that our definition is of independent use by using it to prove the adaptive security of a searchable symmetric encryption scheme introduced by Cash et al. [12].

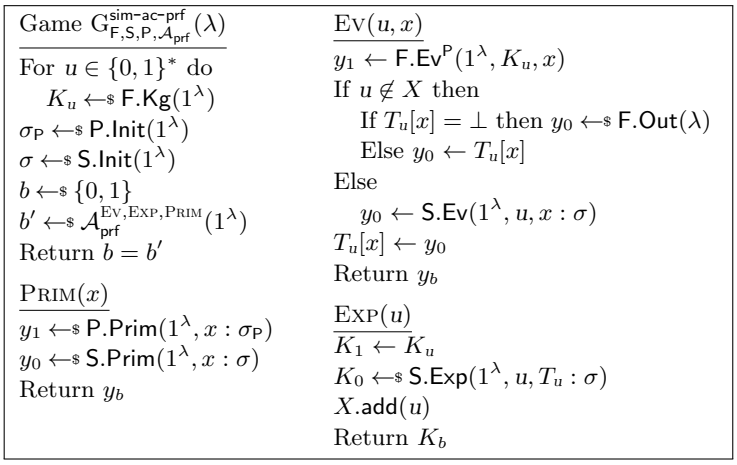

Fig. 6. Game defining multi-user PRF security of F in the face of exposures.

The game $\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{F},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{prf}}}$ is shown in Fig. 6. In the real world, Ev gives adversary $\mathcal{A}_{\mathsf{prf}}$ the real output of F. In the ideal world, Ev's output is chosen at random (and stored in the table T_u), unless u has already been exposed in which case simulator S chooses the output. The table T_u is given to S when an exposure of u happens so it can output a key consistent with prior Ev queries; we assume it is easy to iterate over all $(x, T_u[x])$ pairs for which $T_u[x]$ is not $\perp$. We define $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{F},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{prf}}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{F},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{prf}}}(\lambda)] - 1$. We say F is SIM-AC-PRF secure with P if for all PPT $\mathcal{A}_{\mathsf{prf}}$ there exists a PPT S such that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{F},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{prf}}}(\cdot)$ is negligible.

4 Applications

The value of our definitions stems from their usability in proving the security of protocols constructed from symmetric encryption and pseudorandom functions. In this section, we discuss the application our definitions to simplify and modularize existing security results of Cash et al. [12] and Tyagi et al. [33], and how they imply the notion of equivocable encryption introduced by Jarecki et al. [24].

4.1 Asymmetric Password-Authenticated Key Exchange: OPAQUE

Password-authenticated key exchange (PAKE) protocols allow a client and a server with a shared password to establish a shared key resistant to offline guessing attacks. *Asymmetric* PAKE (aPAKE) further considers security in the case of server compromise, meaning that the server must store some secure representation of the password, rather than the password itself.

OPAQUE [24] is an aPAKE protocol currently being considered for standardization by the IETF. At a high level, OPAQUE is constructed from an oblivious pseudorandom function (OPRF) and an authenticated key exchange protocol (AKE). User key material for the AKE protocol is stored encrypted under an password-derived key from an OPRF. Key exchange proceeds in two steps: (1) the user rederives the encryption key by running the OPRF protocol with the server on their password, then (2) retrieves and decrypts the AKE keys from the server-held ciphertext and proceeds with the AKE protocol. The "commitment problem" arises when an adversary comprises the server state and then later compromises a user password.

$\underline{\text{Game } \mathrm{G}^{\mathsf{eqv}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}}(\lambda)}$	$\underline{\textsc{Prim}(x)}$
$\sigma_{\mathsf{P}} \leftarrow\!\!\$\ \mathsf{P.Init}(1^\lambda)$	$y_1 \leftarrow\!\!\$\ \mathsf{P.Prim}(1^\lambda, x : \sigma_{\mathsf{P}})$
$\sigma \leftarrow\!\!\$\ \mathsf{S.Init}(1^\lambda)$	$y_0 \leftarrow\!\!\$\ \mathsf{S.Prim}(1^\lambda, x : \sigma)$
$b \leftarrow\!\!\$\ \{0,1\}$	Return y_b
$(m, \sigma_{\mathcal{A}}) \leftarrow \mathcal{A}_1^{\textsc{Prim}}(1^\lambda)$	
$K_1 \leftarrow\!\!\$\ \mathsf{SE.Kg}(1^\lambda)$	
$c_1 \leftarrow\!\!\$\ \mathsf{SE.Enc}^{\mathsf{P}}(1^\lambda, K_1, m)$	
$c_0 \leftarrow\!\!\$\ \mathsf{S.Enc}(1^\lambda, \lvert m\rvert : \sigma)$	
$K_0 \leftarrow\!\!\$\ \mathsf{S.Exp}(1^\lambda, m : \sigma)$	
$b' \leftarrow\!\!\$\ \mathcal{A}_2^{\textsc{Prim}}(1^\lambda, c_b, K_b, \sigma_{\mathcal{A}})$	
Return $(b = b')$	

Fig. 7. Game defining EQV security of SE.

Comparison to equivocable encryption. To prove security of their scheme, Jarecki et al. independently propose a weaker version of SIM-AC-CPA that they call equivocable encryption (EQV). Consider game $\mathrm{G}^{\mathsf{eqv}}$ defined in Fig. 7. An encryption scheme SE is *equivocable* if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a simulator S, such that the advantage function $\mathsf{Adv}^{\mathsf{eqv}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{eqv}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}}(\lambda)] - 1$ is negligible. The [24] definition does not specify how to incorporate the ideal model, so we make a reasonable assumption.

Note that EQV is a weaker version of SIM-AC-CPA in that it allows for only one user and only one encryption query. Showing SIM-AC-CPA implies EQV can be done with a simple wrapper reduction in which the output of $\mathcal{A}_1$ from EQV is forwarded to the encryption oracle of SIM-AC-CPA. Since

EQV allows for only one encryption query, we can further show that EQV does not imply SIM-AC-CPA. Consider a scheme that uses a key $K = (K_1, K_2)$ and constructs ciphertexts as $(K_1, \mathsf{Enc}_{K_2}(m))$ unless $m = K_1$, in which case it is formed as $(K_2, \mathsf{Enc}_{K_2}(m))$. Such a scheme could be secure with respect to EQV but will not be secure in a game that allows multiple encryption queries. Interestingly, showing that our multi-user SIM-AC-CPA notion is implied by its single-user version through a hybrid argument is not straightforward due to managing inconsistencies in simulator state between hybrid steps. We have not been able to prove this result and leave it open for future work. Thus, even if EQV was extended to allow multiple encryption queries, it still may not be widely applicable to situations that require multiple users.

Ultimately, our work fills in the claim of Jarecki, et al. that "common encryption modes are equivocable under some idealized assumption".

4.2 Searchable Symmetric Encryption

In the full version of this paper [23], we show that our symmetric encryption and PRF security definitions are useful for proving the security of searchable searchable symmetric encryption (SSE) schemes. An SSE scheme allows a client with a database of documents to store them in encrypted form on a server while still being able to perform keyword searches on these documents.

As a concrete example, we consider Cash et al. [12] which proved non-adaptive security of an SSE scheme when using a PRF and an IND-$ secure encryption scheme and claimed adaptive security when the PRF is replaced with a random oracle and the encryption scheme is replaced with a specific random-oracle-based scheme. We will prove their adaptive result, this time assuming the family of functions is SIM-AC-PRF secure and the encryption scheme is SIM-AC-KP secure. This makes the result more modular because one is no longer restricted to use their specific choices of a PRF and encryption scheme constructed from a random oracle. As a concrete benefit of this, their choice of encryption scheme does not provide INT-CTXT security. To replace the scheme with one that does would require a separate proof while our proof allows the user to choice their favorite INT-CTXT secure scheme without requiring any additional proofs (assuming that scheme is SIM-AC-CPA secure).

Our proof is roughly as complex as their non-adaptive proof; it consists of three similar reductions to the security of the underlying primitives. Without our definitions, a full adaptive proof would have been a technically detailed (though "standard" and not conceptually difficult) proof because it would have to deal with programming the random oracle. Perhaps because of this, the authors of [12] only provided a sketch of the result, arguing that it follows from the same ideas as their non-adaptive proof plus programming the random oracle to maintain consistency. They claim, "[t]he only defects in the simulation occur when an adversary manages to query the random oracle with a key before it is revealed". This is technically insufficient; a defect also occurs if the same key is sampled multiple times by the simulator (analogously to parts of our proofs for Theorem 3 and Theorem 4). In our SSE proof, we need not address these details because

programming the ideal primitive is handled by the assumed simulation security of the underlying primitives.

A large number of other works on SSE have used analogous techniques of constructing a PRF and/or encryption scheme from a random oracle to achieve adaptive security [1, 2, 9, 12, 13, 17, 20, 21, 25–29, 34]. As we discuss in the full version of this paper [23], these papers all similarly elided the details of the random oracle programming proof and/or made mistakes in writing these details. The mistakes are individually small and not difficult to fix, but their prevalence indicates the value our definitions can provide to modularize and simplify the proofs in these works. We chose to analyze the Cash et al. scheme to highlight the application of our definitions because it was the simplest construction requiring both SIM-AC-PRF and SIM-AC-KP secure and because their thorough non-adaptive proof served as a useful starting point from which to build our proof.

4.3 Self-Revocable Encrypted Cloud Storage: BurnBox

In the full version of this paper [23], we consider the BurnBox construction of a self-revocable cloud storage scheme proposed by Tyagi et al. [33]. Its goal is to help provide privacy in the face of an authority searching digital devices, e.g., searches of mobile phones or laptops at border crossings. In their proposed scheme a user stores encrypted version of their files on cloud storage. At any point in time they are able to temporarily revoke their own access to these files. Thereby an authority searching their device is unable to learn the content of these files despite their possession of all the secrets stored on the user's device.

Proving security of their scheme in their security model necessitates solving the "commitment problem." A simulator is forced to simulate the attacker's view by providing ciphertexts for files that it does not know the contents of, then later produce a plausible looking key which decrypts the files properly when told the contents. To resolve this issue in their security they modeled the symmetric encryption scheme in the ideal encryption model (which they introduced for this purpose). We are able to recover their result assuming the SIM-AC-CPA security of the encryption scheme. This provides rigorous justification for the use of practically-used encryption schemes which cannot necessarily be thought of as well modeled by the ideal encryption model (e.g. AES-GCM which they used in their prototype implementation). Moreover, the proof we obtain is simpler than the original proof of Tyagi et al. because we do not have to reason about the programming of the ideal encryption model. The original proof has a bug in this programming which we discuss in the full version of this paper [23].

5 Symmetric Encryption Security Results

In this section, we show that important existing results about the security of symmetric encryption schemes "carry over" to our new definitions. These results (together with our results in the next section) form the foundation of our claim

that encryption schemes used in practice can be considered to achieve SIM-AC-AE security when their underlying components are properly idealized. First, we show that SIM-AC-CPA and INT-CTXT security imply SIM-AC-CCA security. Then we show that the classic Encrypt-then-MAC scheme achieves SIM-AC-CCA security. Each of these results are, conceptually, a straightforward extension of their standard proof. Finally, we show that random oracles and ideal ciphers are SIM-AC-PRF secure and ideal encryption [33] is SIM-AC-AE secure.

CPA and CTXT imply CCA. The following theorem captures that SIM-AC-CPA and INT-CTXT security imply SIM-AC-CCA security. Bellare and Namprempre [8] showed the analogous result for IND-CPA and IND-CCA security.

Theorem 1. *If* SE *is SIM-AC-CPA and INT-CTXT secure with* P*, then* SE *is SIM-AC-CCA secure with* P*.*

Proof (Sketch). Here we sketch the main ideas of the proof. The full details are provided in the full version of this paper [23].

The SIM-AC-CCA simulator we provide is parameterized by a SIM-AC-CPA simulator $\mathsf{S}_{\mathsf{cpa}}$. As state it stores σ of $\mathsf{S}_{\mathsf{cpa}}$ and keeps each K_u that is has returned to exposure queries. For PRIM, ENC, and EXP queries it simply runs $\mathsf{S}_{\mathsf{cpa}}$. For DEC queries it does one of two things. If u has already been exposed it uses the key it previously returned to run the actual decryption algorithm (with oracle access to $\mathsf{S}_{\mathsf{cpa}}$'s emulation of P) and returns the result. Otherwise it assumes the adversary has failed at producing a forgery and simply returns $\perp$. (Note this means we have SIM-AC-AE security if SE is SIM-AC-\$ secure.)

The SIM-AC-CPA security of SE ensures that the adversary cannot differentiate between the real and ideal world queries to PRIM, ENC, and EXP. The INT-CTXT security of SE does the same for the DEC queries. In the full proof we show that $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cca}}_{\mathsf{SE},\mathsf{S}_{\mathsf{cca}},\mathsf{P},\mathcal{A}_{\mathsf{cca}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{int\text{-}ctxt}}_{\mathsf{SE},\mathsf{P},\mathcal{A}_{\mathsf{ctxt}}}(\lambda) + \mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S}_{\mathsf{cpa}},\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\lambda)$. □

Encrypt-then-MAC. Let SE be an encryption scheme. Let F be a family of functions for which $\mathsf{F.Inp}(\lambda) = \{0,1\}^*$. Then the Encrypt-then-MAC encryption scheme using SE and F is denoted $\mathsf{EtM}[\mathsf{SE},\mathsf{F}]$. Its message space is defined as $\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{M}(\lambda) = \mathsf{SE}.\mathsf{M}(\lambda)$. If SE expects access to ideal primitive P_1 and F expects access to ideal primitive P_2 then $\mathsf{EtM}[\mathsf{SE},\mathsf{F}]$ expects access to $\mathsf{P}_1 \times \mathsf{P}_2$. The key-generation algorithm $\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{Kg}$ returns $K = (K_{\mathsf{SE}}, K_{\mathsf{F}})$ where K_{SE} was sampled with $\mathsf{SE}.\mathsf{Kg}(1^\lambda)$ and K_{F} was sampled with $\mathsf{F}.\mathsf{Kg}(1^\lambda)$. Algorithms $\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{Enc}$, and $\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{Dec}$ are defined as follows.

$\underline{\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{Enc}^{\mathsf{P}_1\times\mathsf{P}_2}(1^\lambda, K, m)}$	$\underline{\mathsf{EtM}[\mathsf{SE},\mathsf{F}].\mathsf{Dec}^{\mathsf{P}_1\times\mathsf{P}_2}(1^\lambda, K, (c_{\mathsf{SE}}, \tau))}$
$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow K$	$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow K$
$c_{\mathsf{SE}} \leftarrow\!\!\$\ \mathsf{SE}.\mathsf{Enc}^{\mathsf{P}_1}(1^\lambda, K_{\mathsf{SE}}, m)$	If $\tau \neq \mathsf{F}.\mathsf{Ev}^{\mathsf{P}_2}(1^\lambda, K_{\mathsf{F}}, c_{\mathsf{SE}})$ then return $\perp$
$\tau \leftarrow \mathsf{F}.\mathsf{Ev}^{\mathsf{P}_2}(1^\lambda, K_{\mathsf{F}}, c_{\mathsf{SE}})$	$m \leftarrow \mathsf{SE}.\mathsf{Dec}^{\mathsf{P}_1}(1^\lambda, K_{\mathsf{SE}}, c_{\mathsf{SE}})$
Return (c_{SE}, τ)	Return m

The following theorem establishes that the generic composition result of Bellare and Namprempre [8] holds with our simulation-based definitions of security. We sketch its straightforward proof in the full version of this paper [23].

Theorem 2. *Let* SE *be an encryption scheme. Let* F *be a family of functions for which* $\mathsf{F.Inp}(\lambda) = \{0,1\}^*$*. If* SE *is SIM-AC-CPA secure with* P_1 *and* F *is UF-CMA secure with* P_2*, then* $\mathsf{EtM}[\mathsf{SE},\mathsf{F}]$ *is SIM-AC-CCA secure with* $\mathsf{P}_1 \times \mathsf{P}_2$*.*

Random oracles are good PRFs. We show that a SIM-AC-PRF secure family of functions can be constructed simply in the random oracle model. Consider R defined as follows. It is parameterized by a key-length function $\mathsf{R.kl} : \mathbb{N} \to \mathbb{N}$ and output length function $\mathsf{R.ol} : \mathbb{N} \to \mathbb{N}$. It has input set $\mathsf{R.Inp}(\lambda) = \{0,1\}^*$ and output set $\mathsf{R.Out}(\lambda) = \{0,1\}^{\mathsf{R.ol}(\lambda)}$.

$\underline{\mathsf{R.Kg}(1^\lambda)}$	$\underline{\mathsf{R.Ev}^{\mathsf{P}}(1^\lambda, K, x)}$
$K \leftarrow\!\!\$\ \{0,1\}^{\mathsf{R.kl}(\lambda)}$	$y \leftarrow \mathsf{P}((K \,\|\, x, \mathsf{R.ol}(\lambda)))$
Return K	Return y

Theorem 3. R *is SIM-AC-PRF secure with* $\mathsf{P}_{\mathsf{rom}}$ *if* $\mathsf{R.kl}$ *is super-logarithmic.*

Concretely, in our proof we provide a simulator $\mathsf{S}_{\mathsf{prf}}$ for which we show that,

$$\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{R},\mathsf{S}_{\mathsf{prf}},\mathsf{P}_{\mathsf{rom}},\mathcal{A}_{\mathsf{prf}}}(\lambda) \leq \frac{u_\lambda^2 + p_\lambda u_\lambda}{2^{\mathsf{R.kl}(\lambda)}}$$

where u_λ is an upper bound on the number of users that $\mathcal{A}_{\mathsf{prf}}$ queries to and p_λ is an upper bound on the number of PRIM queries that $\mathcal{A}_{\mathsf{prf}}$ makes.

This theorem captures the random oracle programming implicit in the adaptive security claims of the numerous SSE papers we have identified that used a random oracle like a PRF to achieve adaptive security [1, 2, 9, 12, 13, 17, 20, 21, 25–29, 34]. Of these works, most chose to elide the details of establishing that the adversary cannot detect the random oracle programming, likely considering them simple and/or standard. Despite this, we have identified bugs in all of the proofs that did provide more details. We discuss these bugs in more detail in the full version of this paper [23].

To be clear, we do not claim that any of the SSE schemes studied in these works are insecure. The prevalence of this issue speaks to the difficulty of properly accounting for the details in an ideal model programming proof. Our SIM-AC-PRF notion provides a convenient intermediate definition via which these higher-level protocols could have been proved secure without having to deal with the tedious details of a random oracle programming proof.

Proof (Sketch). Here we sketch the main ideas of the proof. The full details are provided in the full version of this paper [23]. The SIM-AC-PRF simulator works are follows. For PRIM queries it just emulates $\mathsf{P}_{\mathsf{rom}}$ using a table T. For EV queries, it just runs $\mathsf{R.Ev}$ honestly with the key it previously returned for the given user. For EXP queries (on an unexposed user) it picks a random key for this user and sets T to be consistent with values in the table T_u it is given. This simulation is only detectable by an attacker that makes a query to the random oracle with some key that is later chosen by the simulator in response to an

exposure or if the simulator happened to chose the same key for two different users.[6] These events happen with negligible probability. □

Ideal ciphers are good PRFs. One of the most commonly used PRFs is AES so it would be useful to think of it as being SIM-AC-PRF secure; however, due to its invertible nature we cannot realistically model it as a random oracle and refer to the above theorem. Instead, AES is often modeled as an ideal cipher. Let $\mathsf{B.kl} : \mathbb{N} \to \mathbb{N}$ be given and consider B defined as follows. It has input set $\mathsf{B.Inp}(\lambda) = \{0,1\}^{n(\lambda)}$ and output set $\mathsf{B.Out}(\lambda) = \{0,1\}^{n(\lambda)}$.

$\underline{\mathsf{B.Kg}(1^\lambda)}$	$\underline{\mathsf{B.Ev}^{\mathsf{P}}(1^\lambda, K, x)}$
$K \leftarrow\!\!\$\ \{0,1\}^{\mathsf{B.kl}(\lambda)}$	$y \leftarrow \mathsf{P}((+, K, x))$
Return K	Return y

The following establishes that an ideal cipher is SIM-AC-PRF secure.

Theorem 4. B *is SIM-AC-PRF secure with* $\mathsf{P}^n_{\mathsf{icm}}$ *if* $\mathsf{B.kl}$*, n are super-logarithmic.*

Concretely, in our proof we provide a simulator $\mathsf{S}_{\mathsf{prf}}$ for which we show that,

$$\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{B},\mathsf{S}_{\mathsf{prf}},\mathsf{P}^n_{\mathsf{icm}},\mathcal{A}_{\mathsf{prf}}}(\lambda) \leq \frac{u_\lambda^2 + p_\lambda u_\lambda}{2^{\mathsf{B.kl}(\lambda)}} + \frac{q_\lambda^2}{2^{n(\lambda)+1}}$$

where u_λ is an upper bound on the number of users that $\mathcal{A}_{\mathsf{prf}}$ queries to, p_λ is an upper bound on the number of Prim queries that $\mathcal{A}_{\mathsf{prf}}$ makes, and q_λ is an upper bound on the number of Ev queries that $\mathcal{A}_{\mathsf{prf}}$ makes.

The proof of this theorem follows the same general pattern as the proof that a random oracle is SIM-AC-PRF secure (Theorem 3). It only needs to extend the ideas of this prior result slightly to apply a birthday bound so that we can treat the values of $\mathsf{P}^n_{\mathsf{icm}}$ as being sampled with replacement. It works best to process this step last so we do not have to consider the order in which queries are made. The proof is given in the full version of this paper [23].

Ideal encryption model. In the full version of this paper [23], we recall the ideal encryption model used in the analysis of Tyagi et al. [33] and show that it gives a SIM-AC-AE secure encryption scheme. While doing so, we identify and show how to fix a bug in their proof which used this model.

6 Security of Modes of Operation

In the previous section, we showed that existing analysis of the integrity of a symmetric encryption scheme carries over to our simulation setting to lift SIM-AC-CPA security to SIM-AC-CCA security. It would be convenient to be able to similarly prove that existing IND-CPA security of an encryption scheme suffices

[6] The latter of these points is the subtle issue that does not have appear to have been identified in *any* of the SSE papers that were (implicitly) using a random oracle as a SIM-AC-PRF.

to imply SIM-AC-CPA security. Unfortunately, we cannot possibly hope for this to be the case. We know that IND-CPA security can be achieved in the standard model (assuming one-way functions exist), but SIM-AC-CPA security necessarily requires the use of ideal models.

For any typical encryption scheme we could figure out the appropriate way to idealize its underlying components and then write a programming proof to establish security. This would likely be detail intensive and prone to mistakes. We can improve on this by noting that typical symmetric encryption schemes are built as modes of operation using an underlying PRF. We can aim to prove security more modularly by assuming the SIM-AC-PRF security of the underlying family of functions. This alleviates the detail-intensiveness of the proof because the ideal model programming has already been handled in the assumption of SIM-AC-PRF security; it can simply be "passed" along to the new analysis.

In this section, we will show that we can do *even better* than that. We will restrict attention to modes of operation which are IND-$ secure when built from a PRF and satisfy a special extractability property we define in Section 6.1 (which standard examples of models of operation do). Then, in Section 6.2, we establish a generic proof framework to elevate an existing IND-$ security proof to a SIM-AC-$ security proof, by showing that existing proofs of IND-$ security security tend to (implicitly) prove that the scheme satisfies our extractability property. Finally, in Section 6.3 we discuss how the techniques of this section can be extended to other constructions not captured by our formalism, but also note the existence of a (contrived) mode of operation which is IND-$ secure with any secure PRF, but is never SIM-AC-$ secure.

6.1 Modes of Operation and Extractability

We first need to have a formalism capturing what a mode of operation is. Our formalism does not capture all possible modes of operation, but does seem to capture most constructions that are of practical interest and would not be hard to modify to capture other constructions.

A mode of operation SE specifies efficient algorithms $\mathsf{SE.Kg}$, $\mathsf{SE.Enc}$, and $\mathsf{SE.Dec}$ as well as sets $\mathsf{SE.M}$, $\mathsf{SE.Out}$, $\mathsf{SE.FInp}$, and $\mathsf{SE.FOut}$. For any family of functions F with $\mathsf{F.Inp} = \mathsf{SE.FInp}$ and $\mathsf{F.Out} = \mathsf{SE.FOut}$, it defines a symmetric encryption scheme $\mathsf{SE}[\mathsf{F}]$ as follows.

$\underline{\mathsf{SE}[\mathsf{F}].\mathsf{Kg}(1^\lambda)}$	$\underline{\mathsf{SE}[\mathsf{F}].\mathsf{Enc}^{\mathsf{P}}(1^\lambda, K, m)}$	$\underline{\mathsf{SE}[\mathsf{F}].\mathsf{Dec}^{\mathsf{P}}(1^\lambda, K, c)}$
$K_{\mathsf{F}} \leftarrow\!\!\$\ \mathsf{F.Kg}(1^\lambda)$	$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow K$	$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow K$
$K_{\mathsf{SE}} \leftarrow\!\!\$\ \mathsf{SE.Kg}(1^\lambda)$	$c \leftarrow\!\!\$\ \mathsf{SE.Enc}^{\mathsf{F}^{\mathsf{P}}_{K_{\mathsf{F}}}}(1^\lambda, K_{\mathsf{SE}}, m)$	$m \leftarrow \mathsf{SE.Dec}^{\mathsf{F}^{\mathsf{P}}_{K_{\mathsf{F}}}}(1^\lambda, K_{\mathsf{SE}}, c)$
Return $(K_{\mathsf{SE}}, K_{\mathsf{F}})$	Return c	Return m

The superscript $\mathsf{F}^{\mathsf{P}}_{K_{\mathsf{F}}}$ is shorthand for oracle access to $\mathsf{F.Ev}^{\mathsf{P}}(1^\lambda, K_{\mathsf{F}}, \cdot)$. It is required that $\mathsf{SE}[\mathsf{F}].\mathsf{M} = \mathsf{SE.M}$. Moreover, for a given $\lambda \in \mathbb{N}$ the encryption of a message $m \in \mathsf{SE.M}(\lambda)$ must always be in $\mathsf{SE.Out}(\lambda, |m|)$.

Suppose we want to prove that SE is SIM-AC-$ whenever F is SIM-AC-PRF. The natural way to do so is to build our simulator S from the encryption scheme

from the given simulator $\mathsf{S_F}$ for F. In PRIM we can simply have $\mathsf{S.Prim}$ run $\mathsf{S_F.Prim}$. In ENC the ciphertext is chosen at random if the user has not been exposed, otherwise we can simply run $\mathsf{SE.Enc}$ but use $\mathsf{S_F.Ev}$ in place of $\mathsf{F}_{K_\mathsf{F}}$. This just leaves EXP, here we are given a list of ciphertexts for the user and need to output a key to "explain" them. A natural approach is to randomly pick our own K_SE and use $\mathsf{S_F.Exp}$ to chose K_F. Doing so requires giving $\mathsf{S_F}$ a list of input and outputs to the family of function. Intuitively, it seems we want to be able to "extract" a list of input-outputs pairs for F that explain our ciphertexts.

Extractability. A mode of operation is extractable if it additionally specifies an efficient extraction algorithm $\mathsf{SE.Ext}$ satisfying a correctness and uniformity property we now define. The extraction algorithm $\mathsf{SE.Ext}$ has syntax $(\boldsymbol{y}, r) \leftarrow\!\!\$\ \mathsf{SE.Ext}(1^\lambda, K_\mathsf{SE}, c, m)$. The goal of this algorithm is to "extract" a sequence of responses $\boldsymbol{y}$ by F and a string of randomness r that explains how message m could be encrypted to ciphertext c when using key K_SE. We formally define correctness by the following game. It is assumed that $\mathsf{SE.Ext}$ provides outputs of the appropriate lengths to make this code well-defined. Extraction correctness of SE requires that $\Pr[\mathrm{G}^{\mathsf{corr}}_{\mathsf{SE},m}(1^\lambda)] = 1$ for all $\lambda \in \mathbb{N}$ and $m \in \mathsf{SE.M}(\lambda)$.

Game $\mathrm{G}^{\mathsf{corr}}_{\mathsf{SE},m}(1^\lambda)$	Distribution 1
$K_\mathsf{SE} \leftarrow\!\!\$\ \mathsf{SE.Kg}(1^\lambda)$	$c \leftarrow\!\!\$\ \mathsf{SE.Out}(\lambda, \lvert m\rvert)$
$c \leftarrow\!\!\$\ \mathsf{SE.Out}(\lambda, \lvert m\rvert)$	$(\boldsymbol{y}, r) \leftarrow\!\!\$\ \mathsf{SE.Ext}(1^\lambda, K_\mathsf{SE}, c, m)$
$(\boldsymbol{y}, r) \leftarrow\!\!\$\ \mathsf{SE.Ext}(1^\lambda, K_\mathsf{SE}, c, m)$	Return $(\boldsymbol{y}, r)$
$i \leftarrow 0$	Distribution 2
$c' \leftarrow \mathsf{SE.Enc}^{\mathrm{RF}}(1^\lambda, K_\mathsf{SE}, m; r)$	For $i = 1, \ldots, q(\lambda, \lvert m\rvert)$ do
Return $c = c'$	$\quad \boldsymbol{y}[i] \leftarrow\!\!\$\ \mathsf{SE.Out}(\lambda)$
$\mathrm{RF}(x)$	$r \leftarrow\!\!\$\ \{0,1\}^{l(\lambda, \lvert m\rvert)}$
$i \leftarrow i + 1$	Return $(\boldsymbol{y}, r)$
Return $\boldsymbol{y}[i]$	

We will also require a uniformity property of $\mathsf{SE.Ext}$. Specifically we require that its output be uniformly random whenever c is. Formally, there must exist $q, l : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that the two distributions on the right above are equivalent for all $\lambda \in \mathbb{N}$, $m \in \mathsf{SE.M}(\lambda)$, and $K_\mathsf{SE} \in [\mathsf{SE.Kg}(1^\lambda)]$.[7]

Extraction security. A core step in our proof will require an additional property of SE which we will now define. Roughly, the desired property is that if $\mathsf{SE.Ext}$ is repeatedly used to explain randomly chosen ciphertexts an adversary cannot notice if it causes inconsistent values to be returned to $\mathsf{SE.Enc}$.

Formally, consider the game $\mathrm{G}^{\mathsf{ind\text{-}ac\text{-}ext}}$ shown in Fig. 8. In it, a key is chosen for each user and then the adversary is given access to an encryption oracle. In this oracle a random ciphertext is sampled. Then $\mathsf{SE.Ext}$ is run to provide vector $\boldsymbol{y}$ and coins r which explain this ciphertext with respect to the queried message. Finally, $\mathsf{SE.Enc}$ is run with coins r and access to an oracle RF whose behavior depends on the chosen $\boldsymbol{y}$. The ciphertext it outputs is returned to the adversary.

[7] Computational relaxations of our uniformity and correctness property would suffice for our results, but seem to be unnecessary for any "natural" modes of operation.

$\underline{\text{Game } \mathrm{G}^{\mathsf{ind\text{-}ac\text{-}ext}}_{\mathsf{SE},\mathcal{A}}(\lambda)}$	$\underline{\textsc{Enc}(u, m)}$	$\underline{\textsc{Rf}(u, x)\ //\textit{private}}$
For $u \in \{0,1\}^*$ do	Require $m \in \mathsf{SE.M}(\lambda)$	$i \leftarrow i + 1$
$\quad K_{\mathsf{SE},u} \leftarrow\!\!\$\ \mathsf{SE.Kg}(1^\lambda)$	Require $u \notin X$	If $T_u[x] \neq \perp$ then
$b \leftarrow\!\!\$\ \{0,1\}$	$c \leftarrow\!\!\$\ \mathsf{SE.Out}(\lambda, \lvert m\rvert)$	$\quad$ If $b = 1$ then
$b' \leftarrow\!\!\$\ \mathcal{A}^{\textsc{Enc},\textsc{Exp}}(1^\lambda)$	$(\boldsymbol{y}, r) \leftarrow\!\!\$\ \mathsf{SE.Ext}(1^\lambda, K_{\mathsf{SE},u}, c, m)$	$\quad\quad \boldsymbol{y}[i] \leftarrow T_u[x]$
Return $(b = b')$	$i \leftarrow 0$	$T_u[x] \leftarrow \boldsymbol{y}[i]$
$\underline{\textsc{Exp}(u)}$	$c \leftarrow \mathsf{SE.Enc}^{\textsc{Rf}(u,\cdot)}(1^\lambda, K_{\mathsf{SE},u}, m; r)$	Return $T_u[x]$
$X.\mathsf{add}(u)$	Return c	
Return $(K_{\mathsf{SE},u}, T_u)$		

Fig. 8. Game defining IND-AC-EXT security of SE. Note that the adversary is not given oracle access to the "private" oracle $\textsc{Rf}$.

When $b = 0$, this oracle simply returns the entries of $\boldsymbol{y}$, one at a time. The value returned for an input x is stored as $T_u[x]$. The behavior when $b = 1$ is similar except that if an input x to $\textsc{Rf}$ is ever repeated for a user u, then the value stored in $T_u[x]$ is used instead of the corresponding entry of $\boldsymbol{y}$. The attacker's goal is to distinguish between these two cases.

The adversary may choose to expose any user u, learning $K_{\mathsf{SE},u}$ and T_u. After doing so it is no longer able to make $\textsc{Enc}$ queries to that user (as captured by the second "Require" statement in $\textsc{Enc}$). Note that by the uniformity of $\mathsf{SE.Ext}$ we could instead think of $\boldsymbol{y}$ and r as simply being picked at random without $\mathsf{SE.Ext}$ being run, but we believe the current framing is conceptually more clear.

We define $\mathsf{Adv}^{\mathsf{ind\text{-}ac\text{-}ext}}_{\mathsf{SE},\mathcal{A}}(\lambda) = 2\Pr[\mathrm{G}^{\mathsf{ind\text{-}ac\text{-}ext}}_{\mathsf{SE},\mathcal{A}}] - 1$ and say that SE is IND-AC-EXT secure if $\mathsf{Adv}^{\mathsf{ind\text{-}ac\text{-}ext}}_{\mathsf{SE},\mathcal{A}}(\cdot)$ is negligible for all PPT $\mathcal{A}$. This notion will be used for an important step of the coming security proof. Of the properties required from an extraction algorithm it is typically the most difficult to verify.

Example Modes. As a simple example, we can consider counter-mode encryption. Let $\mathsf{CTR.ol}, \mathsf{CTR.il} : \mathbb{N} \to \mathbb{N}$ be fixed and the latter be super-logarithmic. Then CTR is defined as follows. Its key generation algorithm, $\mathsf{CTR.Kg}$, always returns ε. Its sets are defined by

$$\mathsf{CTR.M}(\lambda) = (\{0,1\}^{\mathsf{CTR.ol}(\lambda)})^*,\ \ \mathsf{CTR.Out}(\lambda, l) = \{0,1\}^{l+\mathsf{CTR.il}(\lambda)}$$
$$\mathsf{CTR.FInp}(\lambda) = \{0,1\}^{\mathsf{CTR.il}(\lambda)},\ \ \mathsf{CTR.FOut}(\lambda) = \{0,1\}^{\mathsf{CTR.ol}(\lambda)}.$$

Algorithms $\mathsf{CTR.Enc}$, $\mathsf{CTR.Dec}$, and $\mathsf{CTR.Ext}$ are defined below where $+$ is addition modulo $2^{\mathsf{CTR.il}(\lambda)}$ with elements of $\{0,1\}^{\mathsf{CTR.il}(\lambda)}$ interpreted as integers.

$\underline{\mathsf{CTR.Enc}^O(1^\lambda, K_{\mathsf{SE}}, m)}$	$\underline{\mathsf{CTR.Dec}^O(1^\lambda, K_{\mathsf{SE}}, c)}$	$\underline{\mathsf{CTR.Ext}(1^\lambda, K, c, m)}$
$c_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{CTR.il}(\lambda)}$	$c_0 \,\|\, c' \leftarrow c$	$r \leftarrow c_0$
For $i = 1, \ldots, \lvert m\rvert_{\mathsf{CTR.ol}(\lambda)}$	For $i = 1, \ldots, \lvert c'\rvert_{\mathsf{CTR.ol}(\lambda)}$	For $i = 1, \ldots, \lvert m\rvert_{\mathsf{F.ol}(\lambda)}$
$\quad c_i \leftarrow m_i \oplus O(c_0 + i)$	$\quad m_i \leftarrow c_i \oplus O(c_0 + i)$	$\quad \boldsymbol{y}[i] \leftarrow m_i \oplus c_i$
Return c	Return m	Return $(\boldsymbol{y}, r)$

It is clear that CTR.Ext is correct and that its outputs are distributed uniformly when c is picked at random. The IND-AC-EXT security of CTR follows from the probabilistic analysis done in existing proofs of security for CTR, such as the proof of Bellare, Desai, Jokipii, and Rogaway [6]. The standard analysis simply bounds the probability that any of the values $r_1 + 1, \ldots, r_1 + l_1, r_2 + 1, \ldots, r_2 + l_2, \ldots, r_q + 1, \ldots, r_q + l_q$ collide when the r_i's are picked uniformly and the l_i's are adaptively chosen (before the corresponding r_i is chosen).

Other IND-AC-EXT secure modes of operation include cipher-block chaining (CBC), cipher feedback (CFB), and output feedback (OFB).

6.2 Extractability Implies SIM-AC-$ Security

Finally, we can state the main result of this section, that IND-AC-EXT security of an extractable mode of operation implies SIM-AC-$ security.

Theorem 5. *Let* SE *be an extractable mode of operation which is IND-AC-EXT secure. Then* $\mathsf{SE}[\mathsf{F}]$ *is SIM-AC-$ secure with* P *whenever* F *is SIM-AC-PRF secure with* P *and satisfies* $\mathsf{F.Inp} = \mathsf{SE.FInp}$ *and* $\mathsf{F.Out} = \mathsf{SE.FOut}$.

The full proof is given in the full version of this paper [23]. It considers a sequence of games which transition from the real world of $\mathrm{G}^{\mathsf{sim\text{-}ac\text{-}cpa}}$ to the ideal world (using a simulator we specify). In the first transition we use the security of F to replace SE's oracle access to it with oracle access to a lazily-sampled random function (or simulation by a given simulator $\mathsf{S}_{\mathsf{prf}}$ if the corresponding user has been exposed). Next we modify the game so that (for unexposed users) ciphertexts are chosen at random and then explained by SE.Ext. Then SE.Enc is run with the chosen random coins and oracle access to this explanation (except for whenever a repeat query is made) to produce a modified ciphertext which is returned. The uniformity of SE.Ext ensures this game is identical to the prior game. Then we apply the IND-AC-EXT security of SE so that the oracle given to SE.Enc is not kept consistent on repeated queries. The correctness of SE.Ext gives that the output of SE.Enc is equal to the c that was sampled at random. We provide simulator $\mathcal{S}_{\$}$ that simulates this game perfectly. It runs $\mathsf{S}_{\mathsf{prf}}$ whenever the game would. On an exposure it generate the table T_u for $\mathsf{S}_{\mathsf{prf}}$ by running SE.Ext on ciphertexts to obtain explanatory outputs of the PRF.

Concretely, in the proof we construct adversaries $\mathcal{A}_{\mathsf{prf}}$ and $\mathcal{A}_{\mathsf{ext}}$ along with simulator $\mathsf{S}_{\mathsf{cpa}}$ for which we show

$$\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE}[\mathsf{F}],\mathsf{S}_{\$}[\mathsf{S}_{\mathsf{cpa}}],\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}prf}}_{\mathsf{F},\mathsf{S}_{\mathsf{prf}},\mathsf{P},\mathcal{A}_{\mathsf{prf}}}(\lambda) + \mathsf{Adv}^{\mathsf{ind\text{-}ac\text{-}ext}}_{\mathsf{SE},\mathcal{A}_{\mathsf{ext}}}(\lambda).$$

In the full version of this paper [23], we show that a variant of IND-AC-EXT security without exposures (which we call IND-EXT) necessarily holds if $\mathsf{SE}[\mathsf{F}]$ is single-user IND-$ secure for all single-user PRF secure F's. Moreover, we identify that the typical way that IND-EXT security is shown in security proofs for SE is by proving a slightly stronger property which *will* suffice to imply IND-AC-EXT security. Thereby, one can obtain a SIM-AC-$ security proof from a IND-$ security proof by using the information theoretic core of the existing proof.

6.3 Extensions and a Counter-example Construction

Simple extensions. For encryption schemes not covered by our formalism, it will often be easy to extend the underlying ideas to cover the scheme. Suppose SE uses two distinct function families as PRFs, one could extend our mode of operation syntax to cover this by giving two separate PRF oracles to the encryption and decryption oracles. Then security would follow if there is an extraction algorithm satisfies analogous properties which explains outputs for both of the oracles. The proof would just require an additional step in which the second SIM-AC-PRF is replaced with simulation, as in our transition between games G_0 and G_1.

One can analogously prove the SIM-AC-$ security of the Encrypt-then-PRF construction, where instead of a second SIM-AC-PRF function family we have a SIM-AC-$ encryption scheme. From random ciphertexts it is straightforward to extract the required output of the function family and encryption scheme.

We can also extend the analysis to cover GCM when its nonces chosen uniformly at random. It is not captured by our current syntax because the encryption algorithm always applies the PRF to the all-zero string to derive a sub-key for a hash function. It is straightforward to extend our extraction ideas to allow consistency on this PRF query while maintaining our general proof technique.

Non-extractable counterexample. We showed our general security result for extractable modes of operations and described how to extend it for some simple variants. One might optimistically hope that SIM-AC-$ security would hold for any IND-$ secure mode of operation (when a SIM-AC-PRF secure function family is used). Unfortunately, we can show that this is not the case. We can provide an example mode of operation which is IND-$ secure when using a PRF, but not SIM-AC-CPA secure for any choice of function family. It will be clear that this mode of operation is not extractable, as required by our earlier theorem.

Fix $n : \mathbb{N} \to \mathbb{N}$. Let G be a function family that is OW secure with P_{sm} and for which $\mathsf{G.Kg}(1^\lambda)$ always returns ε and $\mathsf{G.Ev}(1^\lambda, \varepsilon, \cdot)$ is always a permutation on $\{0,1\}^{n(\lambda)}$. Such a G us a one-way permutation on n-bits. From G we construct our counterexample CX. It has sets $\mathsf{CX.Out}(\lambda, l) = \{0,1\}^{l+n(\lambda)}$ and $\mathsf{CX.M}(\lambda) = \mathsf{CX.FInp}(\lambda) = \mathsf{CX.FOut}(\lambda) = \{0,1\}^{n(\lambda)}$. Key generation is given by $\mathsf{CX.Kg} = \mathsf{G.Kg}$. Encryption and decryption are given as follows.

$\underline{\mathsf{CX.Enc}^O(1^\lambda, K_{\mathsf{SE}}, m)}$	$\underline{\mathsf{CX.Dec}^O(1^\lambda, K_{\mathsf{SE}}, c)}$
$c_0 \leftarrow\!\!\$\ \{0,1\}^{n(\lambda)}$	$c_0 \,\|\, c_1 \leftarrow c$
$y \leftarrow \mathsf{G.Ev}^\varepsilon(1^\lambda, \varepsilon, O(c_0))$	$y \leftarrow \mathsf{G.Ev}^\varepsilon(1^\lambda, \varepsilon, O(c_0))$
$c_1 \leftarrow y \oplus m$	$m \leftarrow y \oplus c_1$
Return c	Return m

Above, the superscript ε is used as shorthand for the oracle that always returns ε. Note that this is exactly the behavior of G's expected ideal primitive P_{sm}. This counterexample uses the ideas originally introduced by Fischlin et al. [18] to construct non-programmable random oracles by exploiting a one-way permutation. The construction is not extractable because doing so would require being able

to invert the one-way permutation. The following theorem formally establishes that this is a counterexample.

Theorem 6. *Fix $n : \mathbb{N} \to \mathbb{N}$. Let G be a one-way permutation on n-bits. Let F be a family of functions with $\mathsf{F.Out}(\lambda) = \mathsf{F.Inp}(\lambda) = \{0,1\}^{n(\lambda)}$ and P be an ideal primitive. Then $\mathsf{CX}[\mathsf{F}]$ is IND-\$ secure with P if F is PRF secure with P. However, $\mathsf{CX}[\mathsf{F}]$ is not SIM-AC-CPA secure with P.*

Proof (Sketch). That $\mathsf{CX}[\mathsf{F}]$ is IND-\$ secure when F is PRF secure follows from, e.g., the standard security proof for CTR plus the observation that a permutation applied to a PRF is still a PRF. For the negative result, let S be any simulator and consider the following SIM-AC-CPA adversary $\mathcal{A}_{\mathsf{cpa}}$ and OW adversary $\mathcal{A}$.

$\underline{\mathcal{A}_{\mathsf{cpa}}^{\textsc{Enc},\textsc{Exp},\textsc{Prim}}(1^\lambda)}$	$\underline{\mathcal{A}^{\textsc{Prim}}(1^\lambda, K, y)}$
$m \leftarrow\!\!\$\ \{0,1\}^{n(\lambda)}$	$\sigma \leftarrow \mathsf{S.Init}(1^\lambda)$
$c_0 \,\|\, c_1 \leftarrow \textsc{Enc}(1, m)$	$c_0 \,\|\, c_1 \leftarrow\!\!\$\ \mathsf{S.Enc}(1^\lambda, 1, n(\lambda) : \sigma)$
$y \leftarrow c_1 \oplus m$	$m \leftarrow c_1 \oplus y$
$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow \textsc{Exp}(1)$	$M.\mathsf{add}(m)\,;\, C.\mathsf{add}(c_0 \,\|\, c_1)$
$x \leftarrow \mathsf{F.Ev}^{\textsc{Prim}}(1^\lambda, K_{\mathsf{F}}, c_0)$	$(K_{\mathsf{SE}}, K_{\mathsf{F}}) \leftarrow\!\!\$\ \mathsf{S.Exp}(1^\lambda, 1, M, C : \sigma)$
If $\mathsf{G.Ev}^{\varepsilon}(1^\lambda, \varepsilon, x) = y$ then return 1	$x \leftarrow \mathsf{F.Ev}^{\mathsf{S.Prim}(1^\lambda, \cdot : \sigma)}(1^\lambda, K_{\mathsf{F}}, c_0)$
Return 0	Return x

Adversary $\mathcal{A}_{\mathsf{cpa}}$ queries for the encryption of a random message. Then it exposes the corresponding users and uses the given key to calculate the input-output pair this claims for G. If indeed, this is a valid pair it returns 1, otherwise it returns 0. When $b = 1$, note that $\mathcal{A}_{\mathsf{cpa}}$ will always return 1. Intuitively, when $b = 0$, adversary $\mathcal{A}_{\mathsf{cpa}}$ should almost never return 1 because from the perspective of the simulator S it looks like y was chosen at random, so finding a pre-image for it requires breaking the security of G.

This intuition is captured by the adversary $\mathcal{A}$. It simulates the view S would see when run for $\mathcal{A}$, except instead of picking m at random it waits until after running $\mathsf{S.Enc}$ and sets $m \leftarrow c_1 \oplus y$ where y is the G image it was given as input. Note that y is a uniformly random string because G is a permutation and S is only given the length of the message at this point. Thus, this re-ordering of the calculation of m does not change the view of S. By asking S for the appropriate key and running $\mathsf{F.Ev}$, the adversary obtains a potential pre-image for y.

Simple calculations give $\mathsf{Adv}^{\mathsf{sim\text{-}ac\text{-}cpa}}_{\mathsf{SE},\mathsf{S},\mathsf{P},\mathcal{A}_{\mathsf{cpa}}}(\lambda) = 1 - \mathsf{Adv}^{\mathsf{ow}}_{\mathsf{G},\mathsf{P},\mathcal{A}}(\lambda)$. The latter advantage is negligible from the security of G, so the former is non-negligible. □

Extensions to PRFs. It is often useful to construct a PRF H with large input domains from a PRF F with smaller input domains. The smaller PRF F is often thought of as being reasonably modeled by a random oracle or ideal cipher. If the larger construction H is an indifferentiable construction of a random oracle [14, 30], then we can apply Theorem 3 to obtain the SIM-AC-PRF security of H.

In the case that H is not indifferentiable, one can often use techniques similar to the above to lift a PRF security proof for H to a SIM-AC-PRF security proof for H whenever F is SIM-AC-PRF secure. Implicit in the existing security proof

there will often be a way of "explaining" a random output of H with random outputs by F. On exposure queries, the simulator for H would extract these explanations and feed them to the existing simulator for F to obtain the key to output. For primitive queries, it would just run the F simulator and for evaluation queries after exposure it would just run H using the F simulator in place of F.

Acknowledgments. We thank Thomas Ristenpart for insightful discussions and helpful contributions in the earlier stage of this project. Jaeger was supported in part by NSF grant CNS-1717640, NSF grant CNS-1719146, and a Sloan Research Fellowship. Tyagi was supported in part by NSF grant CNS-1704296.

References

1. G. Asharov, M. Naor, G. Segev, and I. Shahaf. Searchable symmetric encryption: optimal locality in linear space via two-dimensional balanced allocations. In D. Wichs and Y. Mansour, editors, *48th ACM STOC*, pages 1101–1114, Cambridge, MA, USA, June 18–21, 2016. ACM Press.
2. G. Asharov, G. Segev, and I. Shahaf. Tight tradeoffs in searchable symmetric encryption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 407–436, Santa Barbara, CA, USA, Aug. 19–23, 2018. Springer, Heidelberg, Germany.
3. M. Barbosa and P. Farshim. Indifferentiable authenticated encryption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 187–220, Santa Barbara, CA, USA, Aug. 19–23, 2018. Springer, Heidelberg, Germany.
4. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274, Bruges, Belgium, May 14–18, 2000. Springer, Heidelberg, Germany.
5. M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 1–15, Santa Barbara, CA, USA, Aug. 18–22, 1996. Springer, Heidelberg, Germany.
6. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403, Miami Beach, Florida, Oct. 19–22, 1997. IEEE Computer Society Press.
7. M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 1–35, Cologne, Germany, Apr. 26–30, 2009. Springer, Heidelberg, Germany.
8. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545, Kyoto, Japan, Dec. 3–7, 2000. Springer, Heidelberg, Germany.
9. R. Bost. $\Sigma o \phi o \varsigma$: Forward secure searchable encryption. In E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, editors, *ACM CCS 2016*, pages 1143–1154, Vienna, Austria, Oct. 24–28, 2016. ACM Press.

10. R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648, Philadephia, PA, USA, May 22–24, 1996. ACM Press.
11. D. Cash, P. Grubbs, J. Perry, and T. Ristenpart. Leakage-abuse attacks against searchable encryption. In I. Ray, N. Li, and C. Kruegel, editors, *ACM CCS 2015*, pages 668–679, Denver, CO, USA, Oct. 12–16, 2015. ACM Press.
12. D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *NDSS 2014*, San Diego, CA, USA, Feb. 23–26, 2014. The Internet Society.
13. D. Cash and S. Tessaro. The locality of searchable symmetric encryption. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 351–368, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.
14. J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård revisited: How to construct a hash function. In V. Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 430–448, Santa Barbara, CA, USA, Aug. 14–18, 2005. Springer, Heidelberg, Germany.
15. R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In A. Juels, R. N. Wright, and S. De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 79–88, Alexandria, Virginia, USA, Oct. 30 – Nov. 3, 2006. ACM Press.
16. Y. Dodis, T. Ristenpart, J. P. Steinberger, and S. Tessaro. To hash or not to hash again? (In)differentiability results for H^2 and HMAC. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 348–366, Santa Barbara, CA, USA, Aug. 19–23, 2012. Springer, Heidelberg, Germany.
17. M. Etemad, A. Küpçü, C. Papamanthou, and D. Evans. Efficient dynamic searchable encryption with forward privacy. *PoPETs*, 2018(1):5–20, Jan. 2018.
18. M. Fischlin, A. Lehmann, T. Ristenpart, T. Shrimpton, M. Stam, and S. Tessaro. Random oracles with(out) programmability. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 303–320, Singapore, Dec. 5–9, 2010. Springer, Heidelberg, Germany.
19. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
20. F. Hahn and F. Kerschbaum. Searchable encryption with secure and efficient updates. In G.-J. Ahn, M. Yung, and N. Li, editors, *ACM CCS 2014*, pages 310–320, Scottsdale, AZ, USA, Nov. 3–7, 2014. ACM Press.
21. S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, and K. Ren. Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 792–800, April 2018.
22. M. S. Islam, M. Kuzu, and M. Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *NDSS 2012*, San Diego, CA, USA, Feb. 5–8, 2012. The Internet Society.
23. J. Jaeger and N. Tyagi. Handling adaptive compromise for practical encryption schemes. Cryptology ePrint Archive, Report 2020/???, 2020. `http://eprint.iacr.org/2020/???`
24. S. Jarecki, H. Krawczyk, and J. Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 456–486, Tel Aviv, Israel, Apr. 29 – May 3, 2018. Springer, Heidelberg, Germany.

25. S. Kamara and C. Papamanthou. Parallel and dynamic searchable symmetric encryption. In A.-R. Sadeghi, editor, *FC 2013*, volume 7859 of *LNCS*, pages 258–274, Okinawa, Japan, Apr. 1–5, 2013. Springer, Heidelberg, Germany.
26. S. Kamara, C. Papamanthou, and T. Roeder. Dynamic searchable symmetric encryption. In T. Yu, G. Danezis, and V. D. Gligor, editors, *ACM CCS 2012*, pages 965–976, Raleigh, NC, USA, Oct. 16–18, 2012. ACM Press.
27. K. S. Kim, M. Kim, D. Lee, J. H. Park, and W.-H. Kim. Forward secure dynamic searchable symmetric encryption with efficient updates. In B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, editors, *ACM CCS 2017*, pages 1449–1463, Dallas, TX, USA, Oct. 31 – Nov. 2, 2017. ACM Press.
28. J. Li, Y. Huang, Y. Wei, S. Lv, Z. Liu, C. Dong, and W. Lou. Searchable symmetric encryption with forward search privacy. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2019.
29. Q. Liu, Y. Tian, J. Wu, T. Peng, and G. Wang. Enabling verifiable and dynamic ranked search over outsourced data. *IEEE Transactions on Services Computing*, pages 1–1, 2019.
30. U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 21–39, Cambridge, MA, USA, Feb. 19–21, 2004. Springer, Heidelberg, Germany.
31. J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126, Santa Barbara, CA, USA, Aug. 18–22, 2002. Springer, Heidelberg, Germany.
32. P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany.
33. N. Tyagi, M. H. Mughees, T. Ristenpart, and I. Miers. BurnBox: Self-revocable encryption in a world of compelled access. In W. Enck and A. P. Felt, editors, *USENIX Security 2018*, pages 445–461, Baltimore, MD, USA, Aug. 15–17, 2018. USENIX Association.
34. C. Zuo, S. Sun, J. K. Liu, J. Shao, and J. Pieprzyk. Dynamic searchable symmetric encryption schemes supporting range queries with forward (and backward) security. In J. López, J. Zhou, and M. Soriano, editors, *ESORICS 2018, Part II*, volume 11099 of *LNCS*, pages 228–246, Barcelona, Spain, Sept. 3–7, 2018. Springer, Heidelberg, Germany.