Security Against Related Randomness Attacks via Reconstructive Extractors*

Kenneth G. Paterson[1], Jacob C. N. Schuldt[2], Dale L. Sibborn[1], and Hoeteck Wee[3]

[1] Royal Holloway, University of London, UK
[2] National Institute of Advanced Industrial Science and Technology, Japan
[3] École Normale Supérieure, Paris, France

Abstract. This paper revisits related randomness attacks against public key encryption schemes as introduced by Paterson, Schuldt and Sibborn (PKC 2014). We present a general transform achieving security for public key encryption in the related randomness setting using as input any secure public key encryption scheme in combination with an auxiliary-input reconstructive extractor. Specifically, we achieve security in the function-vector model introduced by Paterson *et al.*, obtaining the first constructions providing CCA security in this setting. We consider instantiations of our transform using the Goldreich-Levin extractor; these outperform the previous constructions in terms of public-key size and reduction tightness, as well as enjoying CCA security. Finally, we also point out that our approach leads to an elegant construction for Correlation Input Secure hash functions, which have proven to be a versatile tool in diverse areas of cryptography.

Keywords. public-key encryption, related randomness attacks, auxiliary-inputs, reconstructive extractors, CIS hash functions.

1 Introduction

In recent work, and motivated by numerous practical attacks involving diverse kinds of randomness failure, Paterson, Schuldt and Sibborn [22] introduced *related randomness attacks* against public key encryption schemes. In such an attack, the adversary is able to control the randomness and public keys used during encryption; the security target is that messages encrypted under an honestly generated public key should still remain hidden from the adversary to the maximum extent that this is possible. In the model of Paterson *et al.* [22], the adversary is able to force the encryption scheme to use random values that are related to one another in ways that are specified by functions acting on the randomness space of the scheme. This modelling is inspired by practical attacks like

* The first and third authors were supported by EPSRC grant EP/L018543/1, the second author is supported by JSPS KAKENHI Grant Number 15K16006, and the last author is partially supported by ANR Project EnBiD (ANR-14-CE28-0003)

those by Ristenpart and Yilek in [25], which exploit randomness generation in virtual machines, and extends the *Reset Attack (RA)* setting considered by Yilek in [28]. As demonstrated in [22], it is also connected to other research topics such as security against related key (RKA) attacks and leakage resilience[4].

1.1 The RRA Setting

In the Related Randomness Attack (RRA) setting, the adversary can not only force the reuse of existing random values as in the RA setting, but can also force the use of *functions of* those random values. The extra adversarial power in the RRA setting allows the modelling of reset attacks in which the adversary does not have an exact reset capability, but where the randomness used after a reset is in some way related to that used on previous resets. Such behaviours were observed in the experimental work by Ristenpart *et al.* [25], for example. Via access to an **Enc** oracle, the RRA adversary is able to get arbitrary messages encrypted under arbitrary public keys, using functions ϕ of an initial set of well-distributed but unknown random values. The public keys can even be maliciously generated, and hence, the adversary might know the corresponding private keys. The adversary is tasked with winning an indistinguishability-style game defined via a left-or-right oracle, **LR**, which consistently returns the encryption of either the first or second message of message pairs submitted to the oracle. The encryptions are with respect to an honestly generated target public key pk^*, but again where the adversary can force the use of functions ϕ of the initial random values. When the functions ϕ are limited to coming from some set Φ, we speak of a Φ-restricted adversary.

Because the adversary may know all but one of the private keys, it can check that its challenger is behaving correctly with respect to its encryption queries. Moreover, these queries concern public keys that are outside the control of the challenger. This makes achieving security in the RRA setting technically quite challenging, while practically relevant.

1.2 Previous Results

Paterson *et al.* [22] gave a variety of security models and constructions for PKE secure under related randomness attacks (RRA) in the CPA and CCA settings. As a first contribution, they explored the use of the Random Oracle Model, obtaining necessary and sufficient conditions on the function set Φ that are required to obtain RRA security (these being collision-resistance and output-unpredictability of Φ). They also showed how to transform any PKE scheme `PKE` into a new PKE scheme `Hash-PKE` that is RRA-secure for Φ-restricted adversaries, simply by hashing the random input together with the public key and

[4] See also [22] for an extended discussion of the practical motivation for studying related randomness attacks based on the attack literature as represented by [12, 13, 15, 16, 9, 2, 25, 6, 19, 17, 11, 21, 7].

message during encryption. This construction is closely related to approaches in [25] and [3].

In the standard model, Paterson *et al.* were able to show that any Φ-restricted related key attack-secure PRF (RKA-PRF) can be used to build a RRA-secure PKE scheme for Φ-restricted adversaries, thus transferring security from the RKA setting for PRFs to the RRA setting for PKE. Using the RKA-PRFs currently available in the literature [20, 4, 1] to instantiate this construction, schemes secure for function families Φ consisting of polynomials of bounded degree, can be achieved. However, providing security for function families Φ not enjoying such a convenient algebraic structure would be much more relevant to practical related randomness attacks. But this is a challenging task: in fact, the results by Wichs [27] imply that, for a large class of encryption schemes[5], security for arbitrary function families Φ cannot be shown via a black-box reduction, based on any cryptographic game involving a single-stage adversary (e.g. computational assumptions like DDH, IND-CCA security of a public key encryption scheme, etc.). To obtain further constructions, Paterson *et al.* considered weakened security models; the weakening taking place along two independent dimensions: the degree of control that the adversary enjoys over the public keys under which it can force encryptions for related random values, and the degree of adaptivity it has in the selection of functions $\phi \in \Phi$. More specifically, they considered the situations where:

- The public keys are all honestly generated at the start of the security game, the public keys and all but one of the private keys are then given to the adversary, and the adversary can adaptively specify the functions $\phi \in \Phi$ involved in its queries. This is called the honest-key, related randomness attack (HK-RRA) setting in [22].
- There is no restriction on public keys, but instead of letting the adversary adaptively choose the functions $\phi \in \Phi$, the security game itself is parametrised by a vector of functions $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ that will be used in the attack, and security is required to hold for all choices of $\boldsymbol{\phi}$ from some set $\boldsymbol{\Phi}$. This is called the function-vector, related randomness attack (FV-RRA) setting in [22]. The difference between this setting and the (adaptive) HK-RRA setting is subtle, but note that in the FV-RRA setting, the adversary's choice of ϕ_i cannot depend on the oracles' outputs for the previously used functions $\phi_1, \ldots, \phi_{i-1}$, whereas in the HK-RRA setting, it may.

In the first of these two settings, Paterson *et al.* obtained a generic construction for a scheme achieving HK-RRA security based on combining any PKE scheme with a Correlated-Input Secure (CIS) hash function [14]. However, the then-known instantiations of CIS hash functions only enabled them to obtain selective, HK-RRA security for Φ-restricted adversaries where Φ is a large class of polynomial functions. In view of recent results on RKA-PRFs [1], this construction now appears to be superseded by their earlier generic construction using Φ-restricted RKA-PRFs.

[5] Specifically, Wichs' results apply to encryption schemes which are injective with respect to the used randomness.

In the second of the two settings, they gave a direct construction for a PKE scheme that is FV-RRA-CPA secure solely under the DDH assumption, assuming the component functions ϕ_i of $\boldsymbol{\phi}$ are simultaneously hard to invert on a random input. The scheme is based on a specific PKE scheme of Boneh *et al.* [8] that is secure in the so-called *auxiliary input setting*, wherein the adversary is given a hard-to-invert function of the secret key as part of its input. However, in this setting, only a CPA-secure scheme was given in [22].

1.3 Our Contributions

In this paper, we give a new, general transform for achieving FV-RRA-ATK security for hard-to-invert function families from standard IND-ATK security, where ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$. In fact, the transform works for a stronger notion of FV-RRA-ATK security than was originally introduced in [22]: we will allow an adversary to also manipulate the randomness used for the **LR** queries, instead of being restricted to using only the identity function in such queries. Furthermore, besides yielding schemes secure in the CCA setting, which was left as an open problem in [22], we show that this transform allows us to construct encryption schemes that have tighter security reductions (and are more efficient) than the single FV-RRA-CPA secure scheme that was presented in [22]. As motivation for considering the class of hard-to-invert functions, note that achieving FV-RRA-ATK security for this class would be relevant in modelling the one-way state evolution of a PRNG which has exhausted its entropy pool but which doesn't receive new entropy.

Auxiliary-input reconstructive extractors: Our transform makes use of a technical tool called an *auxiliary-input reconstructive extractor*. Classically, an *extractor* is a function `Ext`, which, given an input and a seed, produces an output that is statistically indistinguishable from elements chosen uniformly at random from some set Σ, provided the input is chosen from a distribution with sufficient min-entropy and the seed is chosen uniformly at random. A *reconstructive* extractor is an extractor with the additional property that, roughly speaking, allows the efficient reconstruction of the input x from any distinguisher $\mathcal{D}$ that successfully distinguishes the output of the extractor from random. This is formalised in terms of the existence of an oracle machine `Rec` outputting x. Then an *auxiliary-input* reconstructive extractor is a reconstructive extractor in which the output still remains indistinguishable when the distinguisher $\mathcal{D}$ is also given access to the output of a leakage function $h(\cdot)$ on input x. Our actual definition (Definition 6) extends this idea further still: the distinguisher $\mathcal{D}$ is given either a set of uniformly random values or the set of outputs of the extractor when evaluated on $\phi(x)$ for all $\phi \in \boldsymbol{\phi}$, where $\boldsymbol{\phi}$ is a vector of functions defined by the game.

Our transform (intuition): Equipped with an auxiliary-input reconstructive extractor, our transform to achieve FV-RRA-ATK security is conceptually simple:

– We append a uniformly random extractor seed s to each public key, resulting in a new public key denoted $\hat{pk}$.
– The encryption algorithm consumes a random value r from some set of bit strings; this is fed into the extractor to create a value $K \leftarrow \texttt{Ext}(r, s)$. This value K used as a key for a Pseudorandom Function (PRF) F to compute $r' \leftarrow F_K(\hat{pk}||m)$ where m is the message to be encrypted. Finally, r' is used as the actual randomness for encryption, and we simply encrypt with the original encryption algorithm.
– Decryption works exactly as in the original decryption algorithm.

Details of the construction are given in Figure 4 in Section 4.

Intuitively, a challenge encryption constructed using randomness value $\phi(r)$ remains secure, since the extractor guarantees an output indistinguishable from random, even when the adversary gains access to encryptions under the related randomness values $\phi'(r)$. Hence, the PRF, which uses the extractor output as a key, will guarantee that independent randomness values are used for different public key and message pairs. In turn, this implies that the adversary is forced to break the security of the underlying PKE scheme to learn anything about the encrypted challenge messages. That this approach attains FV-RRA-ATK security is formally proven in Theorem 1.

Instantiations In Section 5, we consider the instantiation of our transform using the Goldreich-Levin extractor. This provides a particularly neat construction of FV-RRA-ATK-secure PKE in which we start with an IND-ATK-secure scheme and augment it with a simple inner-product computation to prepare the key for the PRF. However, we stress that, given the limited strength of known results for the security of the Goldreich-Levin extractor in the auxiliary input setting [10], our results using this extractor are in turn limited to the original FV-RRA-ATK security model of [22] (i.e. in which the adversary is restricted to using the identity function in its **LR** queries). Still, the schemes obtained from using our transform with this extractor have significant benefits compared to the single concrete FV-RRA-CPA-secure scheme from [22]. For example, we obtain shorter public keys and a tighter security reduction compared to the scheme from [22]. Most importantly, we obtain FV-RRA-CCA security in a completely generic way.

Connection to CIS hash functions: As a final contribution, in Section 6, we explore the connections between auxiliary-input reconstructive extractors and Correlated-Input Secure (CIS) hash functions. The latter were introduced by Goyal *et al.* in [14] and have proven useful in a variety of cryptographic constructions including password-based login, efficient searches on encrypted data and RKA-PRFs. We will show that any reconstructive extractor can be used to construct a secure CIS hash function of a certain type. Specifically, our security definition for CIS hash functions involves functions that are selected from pre-specified sets, as opposed to being adaptively selected as in the strongest (but mostly unachieved) definitions in [14]. Using the Goldreich-Levin extractor once

more provides a construction for CIS hash functions that is exquisitely simple: given key $c \in \mathbb{Z}_p^n$ and input $r \in H^n$ (where H is an arbitrary subset of $\mathbb{Z}_p$), the CIS hash function output is simply:

$$h_c(r) := \langle r, c \rangle$$

where the inner product is evaluated over $\mathbb{Z}_p$.

2 Preliminaries

Notation. Throughout the paper we will use $\lambda \in \mathbb{N}$ to denote the security parameter, which will sometimes be written in its unary representation, 1^λ. We denote by $y \leftarrow x$ the assignment of y to x, and by $s \leftarrow_\$ S$ we denote the selection of an element s uniformly at random from the set S. The notation $[n]$ represents the set $\{1, 2, \ldots, n\}$. For an algorithm A, we denote by $y \leftarrow A(x; r)$ that A is run with input x and random coins r, and that the output is assigned to y.

All our security games and proofs will utilise code-based games and the associated language. Here we briefly recall the basic definitions from Bellare *et al.* in [5]. A game consists of at least two procedures. We begin with **Initialise**, which assigns starting values to all variables and then gives outputs, if there are any, to the adversary. The adversary $\mathcal{A}$ may then submit queries to the oracle procedures, and when $\mathcal{A}$ halts (and possibly outputs a value) the **Finalise** procedure begins. **Finalise** will take the output from $\mathcal{A}$ (if there is one) as its input and will output its own value. The value output by **Finalise** is defined to be the output of the game. We write $\mathbb{P}[G^{\mathcal{A}} \Rightarrow b]$ to denote the probability that game G outputs bit b when run with $\mathcal{A}$. For brevity, in what follows ATK will denote either CPA or CCA, where theorems or statements apply to both games. Any proofs or figures will refer to the CCA setting, but may be easily modified to the CPA case.

Public Key Encryption. We denote a specific PKE scheme by $\mathtt{PKE} = (\mathtt{PKE.K}, \mathtt{PKE.E}, \mathtt{PKE.D})$. All three algorithms are polynomial-time. The randomised key generation algorithm $\mathtt{PKE.K}$ takes the security parameter as its input and outputs a key pair (pk, sk). The encryption algorithm, on input a message $m \in \mathcal{M}$ and a public key pk chooses random coins from $\mathtt{Rnd}$ and uses these coins to output a ciphertext c. The decryption algorithm is deterministic. Its inputs are a private key sk and a ciphertext c. The algorithm either outputs a message m or an error symbol $\perp$. We require the scheme $\mathtt{PKE}$ to satisfy the correctness property. That is, for all $\lambda \in \mathbb{N}$, all all pairs (pk, sk) output by the key generation algorithm, and all messages $m \in \mathcal{M}$, we require that $\mathtt{PKE.D}(sk, \mathtt{PKE.E}(pk, m)) = m$.

Definition 1. *The advantage of an* IND-ATK *adversary $\mathcal{A}$ against a scheme* $\mathtt{PKE}$ *is*

$$\mathbf{Adv}^{\text{ind-atk}}_{\mathtt{PKE},\mathcal{A}}(\lambda) := 2 \cdot \mathbb{P}[\text{IND-ATK}^{\mathcal{A}}_{\mathtt{PKE}}(\lambda) \Rightarrow 1] - 1$$

where game IND-ATK *is shown in Figure 1. A scheme* $\mathtt{PKE}$ *is* IND-ATK *secure if the advantage of any polynomial-time adversary is negligible in the security parameter λ.*

proc. Initialise(λ):
$b \leftarrow_{\$} \{0,1\}$;
$(pk, sk) \leftarrow_{\$} \mathtt{PKE.K}(1^{\lambda})$;
$\mathcal{S} \leftarrow \emptyset$;
return pk

proc. LR(m_0, m_1):
$c \leftarrow_{\$} \mathtt{PKE.E}(pk, m_b)$
$\mathcal{S} \leftarrow \mathcal{S} \cup \{c\}$
return c

proc. Dec(c):
if $c \in \mathcal{S}$, return $\perp$
else return $\mathtt{PKE.D}(sk, c)$

proc. Finalise(b'):
If $b = b'$, return 1

Fig. 1. Game IND-ATK for PKE. (If ATK = CPA, the adversary's access to **proc. Dec** is removed.)

proc. Initialise(λ):
$K \leftarrow_{\$} \mathtt{Keys}_{\lambda}$

proc. Function(x):
return $F(K, x)$

proc. Finalise(b):
return b

proc. Initialise(λ):
FunTab $\leftarrow \emptyset$

proc. Function(x):
if FunTab$[x] = \perp$,
  FunTab$[x] \leftarrow_{\$} \mathtt{Rng}_{\lambda}$
return FunTab$[x]$

proc. Finalise(b):
return b

Fig. 2. Games for PRF security. Game PRFReal is on the left, PRFRand on the right.

Pseudorandom Functions. We recall the standard definition of pseudorandom functions:

Definition 2. *Let* $F : \mathtt{Keys}_{\lambda} \times \mathtt{Dom}_{\lambda} \to \mathtt{Rng}_{\lambda}$ *be a family of functions. The advantage of a* PRF *adversary* $\mathcal{A}$ *against* F *is*

$$\mathbf{Adv}^{\mathrm{prf}}_{F,\mathcal{A}}(\lambda) := \mathbb{P}[\mathrm{PRFReal}^{\mathcal{A}}_{F}(\lambda) \Rightarrow 1] - \mathbb{P}[\mathrm{PRFRand}^{\mathcal{A}}_{\$}(\lambda) \Rightarrow 1]$$

where the games PRFReal *and* PRFRand *are defined in Figure 2. We say* F *is a secure* PRF *family if the advantage of any polynomial-time adversary is negligible in the security parameter* λ*.*

3 Function Vector Related Randomness Security

In this section we recall the FV-RRA-ATK notion of security from [22], and then slightly strengthen this definition to encompass a more general attack.

The FV-RRA-ATK game is designed to capture related randomness attacks, in which the adversary is allowed to obtain challenge encryptions, as well as encryptions for maliciously chosen keys, using related randomness values. This is achieved by giving the adversary access to an encryption oracle **Enc** which enables the adversary to manipulate the random values used for the encryption. More specifically, the standard FV-RRA-ATK security game is parametrised by a vector of functions $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$, where $q := q(\lambda)$ is polynomial in the

proc. Initialise(λ):
$b \leftarrow_{\$} \{0,1\}$;
$(pk^*, sk^*) \leftarrow_{\$} \texttt{PKE.K}(1^\lambda)$;
CoinTab $\leftarrow \emptyset$; $\mathcal{S} \leftarrow \emptyset$;
return pk^*

proc. Dec(c):
if $c \in \mathcal{S}$, then return $\perp$
else return $\texttt{PKE.D}(sk^*, c)$

proc. LR(m_0, m_1, i, j):
If CoinTab$[i] = \perp$,
  CoinTab$[i] \leftarrow_{\$} \texttt{Rnd}$
$r_i \leftarrow$ CoinTab$[i]$
$c \leftarrow \texttt{PKE.E}(pk^*, m_b; \phi_j(r_i))$
$\mathcal{S} \leftarrow \mathcal{S} \cup \{c\}$
return c

proc. Enc(pk, m, i, j):
if CoinTab$[i] = \perp$,
  CoinTab$[i] \leftarrow_{\$} \texttt{Rnd}$
$r_i \leftarrow$ CoinTab$[i]$
$c \leftarrow \texttt{PKE.E}(pk, m; \phi'_j(r_i))$
return c

proc. Finalise(b'):
if $b = b'$, return 1

Fig. 3. Game $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK, where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ and $\boldsymbol{\phi}' = (\phi'_1, \ldots, \phi'_{q'})$. (If ATK = CPA, then the adversary's access to **proc. Dec** is removed.)

security parameter λ, and the adversary may request encryption queries by submitting a tuple of the form (pk, m, i, j) to its **Enc** oracle. This tuple consists of a public key pk, a message m, an index i selecting the random value r_i with which to encrypt, and an index j that selects the function ϕ_j that modifies the randomness r_i before encryption. Hence, the adversary will receive the response $\texttt{PKE.E}(pk, m; \phi_j(r_i))$, where the values r_i are uniform and independent. The adversary may furthermore query a Left or Right (**LR**) oracle with a tuple (m_0, m_1, i). The response of this oracle will be $\texttt{PKE.E}(pk^*, m_b; r_i)$, where pk^* is the target public key and b is a bit, both of which are chosen uniformly and independently during the initialisation stage of the security game. Note that the randomness values r_i used to respond to **LR** queries are uniformly chosen random values. In the CCA version of the game, an adversary can additionally submit ciphertexts c to a decryption oracle **Dec**. The decryption oracle will return $\texttt{PKE.D}(sk^*, c)$ as long as the ciphertext c was not returned by the **LR** oracle. When the adversary has made all its (polynomially many) queries, it will submit a bit b' to a **Finalise** procedure, which represents the adversary's guess for the bit b. The **Finalise** procedure will output 1 (representing an adversarial win) if $b = b'$. The security game for this notion is given in Figure 3.

We will now introduce some new definitions that slightly strengthen the FV-RRA-ATK notion from [22] outlined above. Our strengthening allows an adversary to manipulate the randomness used for the **LR** queries, instead of being restricted to using only the identity function. The security game for our new notion is given in Figure 3. The major difference from the definition of [22] is that the game is parametrised by two sets of functions, $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$. An adversary may only use functions from $\boldsymbol{\phi}$ in its **LR** queries, and the functions in $\boldsymbol{\phi}'$ may only be used for **Enc** queries. Notice that if $\boldsymbol{\phi} = \{\text{id}\}$, then this definition recovers the corresponding FV-RRA-ATK security game and notion from [22]. While our generic transform is proven secure in the stronger model shown in Figure 3, we stress that, because of the limitations of currently known reconstructive extractors, our concrete instantiation of the transform will be secure only in the weaker model of [22].

The following definition has been adapted from [22] for our purposes. The definition captures natural restrictions which must be placed on an adversary with the capability of controlling the randomness of the challenge encryptions in an IND-ATK style security game. This is reminiscent of the restrictions put in place in the security definition for deterministic encryption (e.g. see [24]).

Definition 3. *Let $\mathcal{A}$ be an adversary in Game* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *that queries r different randomness indices to its* ***LR*** *and* ***Enc*** *oracles and makes $q_{i,\phi}$ queries to its* ***LR*** *oracle with index i and function $\phi \in \boldsymbol{\phi}$. Let $(m_0^{i,\phi,1}, m_1^{i,\phi,1}), \ldots, (m_0^{i,\phi,q_{i,\phi}}, m_1^{i,\phi,q_{i,\phi}})$ be $\mathcal{A}$'s* ***LR*** *queries for index $i \in [r]$ and $\phi \in \boldsymbol{\phi}$. Suppose that for all pairs $(i, \phi) \in [r] \times \boldsymbol{\phi}$ and for all $j \neq k \in [q_{i,\phi}]$, we have:*

$$m_0^{i,\phi,j} = m_0^{i,\phi,k} \text{ iff } m_1^{i,\phi,j} = m_1^{i,\phi,k}.$$

Then we say that $\mathcal{A}$ is equality-pattern respecting.

Note that any adversary that is not equality-pattern respecting can trivially win the game in Figure 3. More specifically, the adversary can simply query its **LR** oracle with the tuples (m_0, m_1, i, j) and (m_0, m_2, i, j), where m_0, m_1 and m_2 are all distinct. The values i and j can be an arbitrary values from the appropriate domain. If the bit b is equal to 0, the adversary will receive identical ciphertexts, whereas the ciphertexts will differ if b equals 1. This results in a trivial win for an adversary. In contrast, an equality-respecting adversary cannot exploit the available oracles in this particular way, and is forced to mount a non-trivial attack against the scheme to win the security game.

With the above definition in place, we can now formally define FV-RRA-ATK security.

Definition 4. *Let $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ and $\boldsymbol{\phi}' = (\phi'_1, \ldots, \phi'_{q'})$ be vectors of $q := q(\lambda)$ and $q' := q'(\lambda)$ functions respectively. We define the advantage of an equality-pattern respecting,* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary $\mathcal{A}$ against a PKE scheme* PKE *to be:*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-fv-rra-atk}}_{\mathtt{PKE},\mathcal{A}}(\lambda) := 2 \cdot \Pr[(\boldsymbol{\phi}, \boldsymbol{\phi}')\text{-FV-RRA-ATK}^{\mathcal{A}}_{\mathtt{PKE}}(\lambda) \Rightarrow 1] - 1.$$

If $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$ are sets of vectors of functions, then a PKE scheme PKE *is said to be* $(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$-FV-RRA-ATK *secure if, for all $\boldsymbol{\phi} \in \boldsymbol{\Phi}$ and for all $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$, the advantage of any equality-pattern respecting,* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary against* PKE *that runs in polynomial time is negligible in the security parameter λ.*

Similar to the notion defined in [22], it is possible to reduce the above defined FV-RRA-ATK security to a simpler notion in which the security game involves only a single uniformly chosen random value used in all oracle queries. The following lemma follows easily from Lemma 1 of [22] and is therefore presented without a proof.

Lemma 1. *Consider an equality-pattern respecting,* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary $\mathcal{A}$ that queries q_r distinct randomness indices and makes at most q_{LR}*

LR *queries. Then there exists an equality-pattern respecting,* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary* $\mathcal{B}$ *that queries at most* 1 *randomness index and makes at most* q_{LR} ***LR*** *queries such that*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-fv-rra-atk}}_{\mathtt{PKE},\mathcal{A}}(\lambda) \leq q_r \cdot \mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-fv-rra-atk}}_{\mathtt{PKE},\mathcal{B}}(\lambda),$$

where $\mathcal{B}$ *runs in approximately the same time as* $\mathcal{A}$*. In the CCA setting,* $\mathcal{B}$ *makes the same number of decryption queries as* $\mathcal{A}$*.*

4 Obtaining FV-RRA Security from Auxiliary-Input Reconstructive Extractors

In this section we present the main result of the paper. Recall that this result improves upon the work of Paterson *et al.* [22] by proposing a transform that converts any IND-ATK scheme into an FV-RRA-ATK scheme via the use of an auxiliary-input reconstructive extractor. Recall also that the authors of [22] only provided a single concrete instantiation of a FV-RRA-CPA secure scheme. In the later sections we will provide instantiations of our transform that are not only able to meet the stronger FV-RRA-CCA notion, but also provide shorter public keys and a tighter security reduction compared to the scheme from [22].

Before introducing the extractors we utilise in our transform, we first need to define the notion of a vector of functions being δ-hard-to-compute with respect to another vector of functions.

Definition 5. *Let* $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ *and* $\boldsymbol{\phi}' = (\phi'_1, \ldots, \phi'_{q'})$ *denote vectors of functions on a set* $\mathtt{Rnd}_\lambda$*, where* $q := q(\lambda)$ *and* $q' := q'(\lambda)$ *are polynomial in the security parameter* λ*. Let* $\delta(\lambda)$ *be a function. We say that* $\boldsymbol{\phi}$ *is* $\delta(\lambda)$-hard-to-compute with respect to $\boldsymbol{\phi}'$ *if, for all polynomial time algorithms* $\mathcal{A}$ *and all sufficiently large* λ*, we have:*

$$\Pr[\phi_i(r) \leftarrow \mathcal{A}(\phi'_1(r), \ldots, \phi'_q(r)) : r \leftarrow_\$ \mathtt{Rnd}_\lambda] \leq \delta(\lambda),$$

for all $i \in \{1, \ldots, q\}$*. We say that a set of vectors of functions* $\boldsymbol{\Phi}$ *is* δ*-hard-to-compute with respect to* $\boldsymbol{\Phi}'$ *if each vector* $\boldsymbol{\phi} \in \boldsymbol{\Phi}$ *is* δ*-hard-to-compute with respect to every vector in* $\boldsymbol{\Phi}'$ *(note that the vectors in such a set* $\boldsymbol{\Phi}$ *need not all be of the same dimension, but we assume they each have dimension that is polynomial in* λ*). If* $\delta = \mathrm{negl}(\lambda)$*, then we simply say that* $\boldsymbol{\Phi}$ *is hard-to-compute with respect to* $\boldsymbol{\Phi}'$*.*

A natural question to ask is: what functions satisfy this notion of being δ-hard-to-compute? For simplicity, consider the scenario where $\boldsymbol{\Phi} = \{\mathrm{id}\}$ (in which case we simply say that $\boldsymbol{\Phi}'$ is δ-hard-to-invert, cf. Definition 14 of [22]), and assume that $\boldsymbol{\Phi}'$ consists of only one function, say ϕ'. In this scenario, an obvious example of a δ-hard-to-invert function is a function that fixes certain bits of the output e.g. a function ϕ' that takes a bit-string of length n as input, and returns a string consisting of k zero bits followed by the least significant $n-k$ bits of the

input (for $0 \leq k \leq n$). No information is leaked about the first k bits of the input, and hence no algorithm can invert ϕ' with probability greater than 2^{-k} when the input string is uniformly random. Therefore, if $k \geq -\log_2 \delta$, the function ϕ' (and, consequently, $\boldsymbol{\Phi}'$) is δ-hard-to-invert. This example can naturally be extended to the case where $\boldsymbol{\Phi}'$ contains multiple vectors of functions and $\boldsymbol{\Phi} \neq \{\mathrm{id}\}$.

We now introduce our generalised definition of an auxiliary-input reconstructive extractor.

Definition 6. *An* $(\epsilon, \delta, \boldsymbol{\Phi}, \boldsymbol{\Phi}')$-auxiliary-input reconstructive extractor *is a pair of functions* $(\mathtt{Ext}, \mathtt{Rec})$ *such that* $\mathtt{Ext}$ *is an extractor that maps from* $\{0,1\}^n \times \{0,1\}^d$ *to* Σ*, and* $\mathtt{Rec}$ *is an oracle machine that on input* $(1^n, 1/\epsilon)$ *runs in time* $\mathrm{poly}(n, 1/\epsilon, \log(|\Sigma|))$*. Furthermore, for every* $x \in \{0,1\}^n$*, every* $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q) \in \boldsymbol{\Phi}$*, every* $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$*, and every function* $\mathcal{D}$ *such that*

$$\left| \Pr_{s \leftarrow_{\$} \{0,1\}^d} [\mathcal{D}(s, \{\mathtt{Ext}(\phi_i(x), s)\}_{i \in \{1,\ldots,q\}}, \boldsymbol{\phi}'(x)) = 1] - \Pr_{\substack{s \leftarrow_{\$} \{0,1\}^d \\ \sigma_i \leftarrow_{\$} \Sigma}} [\mathcal{D}(s, \{\sigma_i\}_{i \in \{1,\ldots,q\}}, \boldsymbol{\phi}'(x)) = 1] \right| \geq \epsilon$$

we require that

$$\Pr[\mathtt{Rec}^{\mathcal{D}}(1^n, 1/\epsilon, \boldsymbol{\phi}'(x)) = \phi_i(x)] \geq \delta$$

for some $i \in \{1, \ldots, q\}$*, where* $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$*,* $q := q(\lambda)$ *is polynomial, and the probability is over the coin tosses of* $\mathtt{Rec}$*. If, for every* $\mathcal{D}$ *with non-negligible* ϵ*,* $\mathtt{Rec}$ *reconstructs* $\phi_i(x)$ *with non-negligible probability, we may simply say that* $(\mathtt{Ext}, \mathtt{Rec})$ *is a* $(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$*-auxiliary-input reconstructive extractor.*

Armed with this new definition of an auxiliary-input reconstructive extractor, we are ready to state and prove the main result of this paper. We show that any extractor satisfying Definition 6 can be used in conjunction with an IND-ATK secure PKE scheme and a PRF to meet the FV-RRA-ATK security notion in Figure 3. The encryption scheme that achieves this result is in Figure 4. The algorithm works by appending a uniformly random extractor seed to each public key, but leaving the private key unmodified. The encryption algorithm generates a uniformly random r, which is then fed into the extractor (using the seed from the public key). The output of the extractor is used as a key for a PRF, and the input to the PRF is the public key appended with the message. Finally, the output of the PRF is used as the new randomness for encryption, and then we simply encrypt with the standard encryption algorithm.

Theorem 1. *If* $\boldsymbol{\Phi}$ *is hard-to-compute with respect to* $\boldsymbol{\Phi}'$ *and* $(\mathtt{Ext}, \mathtt{Rec})$ *is an* $(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$*-auxiliary-input reconstructive extractor, then the PKE scheme* $\mathtt{EXT\text{-}PKE}$ *in Figure 4 is* $(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$-FV-RRA-ATK *secure when instantiated with a secure PRF and an IND-ATK secure PKE scheme* $\mathtt{PKE}$*. More precisely, consider any polynomial-size vectors of functions* $\boldsymbol{\phi} \in \boldsymbol{\Phi}$ *and* $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$*, any* $(\epsilon, \delta, \boldsymbol{\Phi}, \boldsymbol{\Phi}')$*-auxiliary-input*

Alg. $\texttt{EXT-PKE.K}(1^\lambda)$:	Alg. $\texttt{EXT-PKE.E}(\hat{pk}, m)$:	Alg. $\texttt{EXT-PKE.D}(\hat{sk}, c)$:
$(pk, sk) \leftarrow \texttt{PKE.K}(1^\lambda)$	$r \leftarrow_\$ \texttt{Rnd}$	$m \leftarrow \texttt{PKE.D}(sk, c)$
$s \leftarrow \texttt{seeds}$	$K \leftarrow \texttt{Ext}(r, s)$	return m
$\hat{pk} \leftarrow (pk, s)$	$r' \leftarrow F_K(\hat{pk}\|\|m)$	
$\hat{sk} \leftarrow (sk)$	$c \leftarrow \texttt{PKE.E}(pk, m; r')$	
return $\hat{pk}$	return c	

Fig. 4. Scheme EXT-PKE built from a reconstructive extractor, a PKE scheme PKE, and a PRF F.

reconstructive extractor $(\texttt{Ext}, \texttt{Rec})$*, and any equality-pattern respecting,* $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary* $\mathcal{A}$ *against* EXT-PKE*. Suppose* $\mathcal{A}$ *makes* q_{LR} $\boldsymbol{LR}$ *queries and uses* q_r *randomness indices. Then, either* $\boldsymbol{\Phi}$ *is not* δ*-hard-to-compute with respect to* $\boldsymbol{\Phi}'$*, or there exists a PRF adversary* $\mathcal{B}$*, and an IND-ATK adversary* $\mathcal{C}$*, all running in polynomial time, such that:*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-fv-rra-atk}}_{\texttt{EXT-PKE},\mathcal{A}}(\lambda) < 2q_r \cdot q \cdot \mathbf{Adv}^{\text{prf}}_{F,\mathcal{B}}(\lambda) + q_r \cdot q_{LR} \cdot \mathbf{Adv}^{\text{ind-atk}}_{\texttt{PKE},\mathcal{C}}(\lambda) + 2q_r\epsilon.$$

Proof. See Appendix A.

5 Instantiation of an Auxiliary-Input Reconstructive Extractor

Given Theorem 1, it now remains to see what extractors exist that satisfy Definition 6. The strongest extractor we are aware of is the Goldreich-Levin extractor, whose properties are analysed in [10, Theorem 1]. That theorem states the following (with the notation changed to remain consistent with ours):

Theorem 2. *Let* p *be a prime, and let* H *be an arbitrary subset of* $\mathbb{Z}_p$*. Let* $f : H^n \to \{0,1\}^*$ *be any (possibly randomised) function. If there is a distinguisher* $\mathcal{D}$ *that runs in time* t *such that*

$$\Big|\Pr[\mathbf{r} \leftarrow H^n, y \leftarrow f(\mathbf{r}), \mathbf{s} \leftarrow \mathbb{Z}_p^n : \mathcal{D}(y, \mathbf{s}, \langle \mathbf{r}, \mathbf{s}\rangle) = 1] - \Pr[\mathbf{r} \leftarrow H^n, y \leftarrow f(\mathbf{r}), \mathbf{s} \leftarrow \mathbb{Z}_p^n, u \leftarrow \mathbb{Z}_p : \mathcal{D}(y, \mathbf{s}, u) = 1]\Big| = \epsilon$$

then there is an inverter $\mathcal{A}$ *that runs in time* $t' = t \cdot \text{poly}(n, |H|, 1/\epsilon)$ *such that*[6]

$$\Pr[\mathbf{r} \leftarrow H^n, y \leftarrow f(\mathbf{r}) : \mathcal{A}(y) = \mathbf{r}] \geq \frac{\epsilon^3}{512 \cdot n \cdot p^3}. \tag{1}$$

[6] The bound quoted in [10] had the denominator $512np^2$. However, we believe the bound has a slight error and should in fact be $512np^3$, as given here. The bound in equation (1) was also used by Paterson *et al.* in [23].

Alg. $\texttt{EXT-PKE.K}(1^\lambda)$:	Alg. $\texttt{EXT-PKE.E}(\hat{pk}, m)$:	Alg. $\texttt{EXT-PKE.D}(\hat{sk}, c)$:
$(pk, sk) \leftarrow \texttt{PKE.K}(1^\lambda)$	$r \leftarrow_\$ H^\lambda$	$m \leftarrow \texttt{PKE.D}(sk, c)$
$s \leftarrow \mathbb{Z}_p^\lambda$	$K \leftarrow \langle r, s \rangle$	return m
$\hat{pk} \leftarrow (pk, s)$	$r' \leftarrow F_K(\hat{pk}\|\|m)$	
$\hat{sk} \leftarrow (sk)$	$c \leftarrow \texttt{PKE.E}(pk, m; r')$	
return $\hat{pk}$	return c	

Fig. 5. Scheme EIP-PKE (Euclidean Inner Product) built from a PKE scheme PKE, and a PRF F. Here, H denotes a subset of $\mathbb{Z}_q$.

This theorem can be used to obtain an auxiliary-input reconstructive extractor. Specifically, consider the extractor Ext that maps from $H^n \times \mathbb{Z}_p^n$ to $\mathbb{Z}_p$ (where H is a subset of $\mathbb{Z}_p$) defined as

$$\texttt{Ext}(r, s) = \langle r, s \rangle.$$

Matching the notation of Theorem 2 with Definition 6, Rec is now $\mathcal{A}$, $\boldsymbol{\Phi} = \{\mathrm{id}\}$, $\boldsymbol{\Phi}'$ is the set of δ-hard-to-invert vectors of functions, $\boldsymbol{\phi}'$ is the function f, and the extractor Ext is easily seen to be an $(\epsilon, \delta, \mathrm{id}, \boldsymbol{\Phi}')$-auxiliary-input reconstructive extractor, where

$$\epsilon = \sqrt[3]{512\delta\lambda p^3}.$$

Note that, in the proof of [10], the theorem is stated with one function f. However, we now use a vector of functions $(\phi'_1, \ldots, \phi'_q)$ in our proof. Fortunately this is not problematic, since we can simply interpret f (whose output is in $\{0,1\}^*$) as a vector of functions. That is, we can set $f(r) = (\phi'_1(r), \ldots, \phi'_q(r))$.

By combining Theorem 2 with Theorem 1, we easily obtain the following theorem.

Theorem 3. *Let $\boldsymbol{\Phi}'$ be a set of hard-to-invert vectors of functions on $\{0,1\}^\lambda$. Then PKE scheme* EIP-PKE *in Figure 5 is* $(\mathrm{id}, \boldsymbol{\Phi}')$-FV-RRA-ATK *secure. More precisely, consider any polynomial-size vector of functions $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$ which are δ-hard-to-invert, and any equality-pattern respecting,* $(\mathrm{id}, \boldsymbol{\phi}')$-FV-RRA-ATK *adversary $\mathcal{A}$ against* EIP-PKE*. Suppose $\mathcal{A}$ makes q_{LR} $\boldsymbol{LR}$ queries and uses q_r randomness indices. Then there exists a PRF adversary $\mathcal{B}$ and an IND-ATK adversary $\mathcal{C}$, all running in polynomial time, such that:*

$$\mathbf{Adv}^{(\mathrm{id},\boldsymbol{\phi}')\text{-fv-rra-atk}}_{\texttt{EIP-PKE},\mathcal{A}}(\lambda) < 2q_r \cdot \mathbf{Adv}^{\mathrm{prf}}_{F,\mathcal{B}}(\lambda) + q_r \cdot q_{LR} \cdot \mathbf{Adv}^{\text{ind-atk}}_{\texttt{PKE},\mathcal{C}}(\lambda) + 2q_r \sqrt[3]{512\delta\lambda p^3}.$$

While the above theorem limits the challenge functions modifying the input to the extractor to being the identity function, the schemes resulting from our transform using the above reconstructive extractor still enjoy several advantages over the single FV-RRA-CPA-secure scheme that was presented in [22]. Most notably, [22] only gave one concrete scheme, which is only secure in the CPA version of the FV-RRA-ATK game. Our theorem not only shows how to achieve CCA security (which was left as an open problem in [22]), but also shows how to

convert *any* IND-CCA scheme into an FV-RRA-CCA secure scheme. Furthermore, the security bound of our theorem is tighter than that of [22], and our theorem facilitates the use of much smaller public keys. For comparison, when using our transform with the above Goldreich-Levin extractor, the public key of the underlying PKE scheme is modified to include λ additional components from $H \subset \mathbb{Z}_q$. Hence, transforming, for example, the PKE scheme by Kurosawa and Desmedt [18], yields a scheme with public keys consisting of $\lambda+4$ group elements and a hash function key. In contrast, the modified BHHO scheme presented in [22] requires public keys consisting of $2 \cdot k(\lambda)$ group elements (where k is polynomial). Furthermore, the loss of security in the security reduction of the modified BHHO scheme includes the component $\sqrt[3]{512\delta k p^4}$, which originates from the reduction to the δ-hard-to-invert functions. In comparison, the corresponding loss of security obtained from applying our transform is $\sqrt[3]{512\delta\lambda p^3}$, which leads to a weaker requirement on the δ-hard-to-invert functions.

It remains an open question whether there exists extractors that will enable stronger notions of FV-RRA-ATK security to be shown for schemes like `EXT-PKE` (Figure 4), or alternative extractors that have, for example, shorter seeds. However, this seems difficult at present. A standard technique to obtain an (ϵ, δ)-auxiliary-input reconstructive extractor is to use complexity-leveraging with a standard reconstructive extractor [26]. Unfortunately, this technique does not appear to work in the FV-RRA-ATK setting. More specifically, if we wish to use complexity-leveraging, we require the range of the auxiliary function to be smaller than the domain. However, for our FV-RRA-ATK game to make sense, we require that for each ϕ we have $\mathcal{D}(\phi) = \mathcal{R}(\phi) = \texttt{Rnd}$. Hence, complexity-leveraging seems to be incompatible with the FV-RRA-ATK model.

6 Connections with CIS Hash Functions

We will now briefly explore the connections between $(\epsilon, \delta, \boldsymbol{\Phi}, \boldsymbol{\Phi}')$-auxiliary-input reconstructive extractors and correlated-input secure (CIS) hash functions. In particular, we will show that any reconstructive extractor can be used to construct a secure CIS hash function. Correlated-input secure hash functions were first studied by Goyal *et al.* in [14]. They introduced several definitions of security, but the one we shall be concerned with is the pseudorandomness notion. Intuitively, a hash function is (pseudorandom) correlated-input secure if the challenge output of the hash function is indistinguishable from random even when an adversary is allowed to see outputs on correlated inputs. That is, an adversary can submit correlation functions ϕ to its oracle and will receive $h(\phi(r))$, where h is the (possibly keyed) hash function, and r is a uniformly random input chosen at the beginning of the security game. The adversary may submit multiple oracle queries, and finally forwards a challenge function ϕ^* to the oracle. The game will return either $h(\phi^*(r))$ or z, where z is chosen uniformly at random from the range of the hash function. The hash function is (adaptively) secure if the adversary has negligible advantage in distinguishing the outputs.

proc. Initialise(λ):	**proc. Challenge**(j):	**proc. Query**(i):
$b \leftarrow_{\$} \{0,1\}$;	if $b = 0$,	return $h_c(\phi'_i(r))$
$h_c \leftarrow_{\$} \mathcal{H}$	$z \leftarrow_{\$} \mathcal{R}(h_c)$	
$r \leftarrow_{\$} \mathcal{D}(h_c)$	return z	**proc. Finalise**(b'):
return h_c	else,	If $b = b'$, return 1
	return $h_c(\phi_j(r))$	

Fig. 6. The $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-CIS hash game, where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ and $\boldsymbol{\phi}' = (\phi'_1, \ldots, \phi'_{q'})$.

As noted in [14], CIS hash functions have applications to password-based login and efficient searches on encrypted data. Furthermore, they share interesting connections with Related-Key Attack secure primitives. However, the CIS hash function construction presented in [14] only achieves selective security for correlation functions ϕ corresponding to polynomials of bounded degree, which limits its usefulness in the above mentioned applications. Constructing adaptive CIS hash functions for a wide class of functions is a challenging task, in particular for non-algebraic function classes. This is evidenced by the results of Wichs [27], which show that injective CIS hash functions cannot be proved secure for arbitrary correlation functions ϕ via a black-box reduction, based on any cryptographic game. However, here we show that auxiliary-input reconstructive extractors can be used to construct a specific kind of CIS hash functions. To explore this connection, we must consider a variant of the CIS hash security game that was presented in [14]. The security game is shown in Figure 6, while our definition of security is given below.

Definition 7. *The advantage of an adversary $\mathcal{A}$ against a family of hash functions $\mathcal{H}$ in the $(\boldsymbol{\phi}, \boldsymbol{\phi}')$-CIS game (Figure 6) is defined to be*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-cis}}_{\mathcal{H},\mathcal{A}}(\lambda) := 2 \cdot \Pr[(\boldsymbol{\phi}, \boldsymbol{\phi}')\text{-CIS}^{\mathcal{A}}_{\mathcal{H}}(\lambda) \Rightarrow 1] - 1.$$

Definition 8. *A family of hash functions $\mathcal{H}$ is said to be $(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$-pseudorandom correlated-input secure if, for all $\boldsymbol{\phi} \in \boldsymbol{\Phi}$, all $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$, and all polynomial time adversaries $\mathcal{A}$, we have*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-cis}}_{\mathcal{H},\mathcal{A}}(\lambda) \leq \mathrm{negl}(\lambda).$$

Notice that in our new definition, instead of letting the adversary adaptively choose the functions as in [14], the security game itself is parametrised with function vectors $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$, and security is required to hold for all choices of $\boldsymbol{\phi} \in \boldsymbol{\Phi}$ and $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$. It is worth stressing that there is a subtle difference between the two approaches to defining security for CIS hash functions, and the definition used here implies that the function vectors $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$ will be independent of the chosen hash function (i.e. the hash function key c).

With these definitions and notions in place we can define our hash function family $\mathcal{H}$ from an extractor as follows:

$$h_c(r) := \mathtt{Ext}(r, c).$$

The following theorem establishes the security of the hash function, based on the security of the underlying auxiliary-input reconstructive extractor.

Theorem 4. *Let* $\mathtt{Ext}$ *be an* $(\epsilon, \delta, \boldsymbol{\Phi}, \boldsymbol{\Phi}')$*-auxiliary-input reconstructive extractor, and let* $\boldsymbol{\Phi}$ *be* δ*-hard-to-compute with respect to* $\boldsymbol{\Phi}'$*. Consider the hash function family* $\mathcal{H}$ *defined by the hash functions* $h_c(r) := \mathtt{Ext}(c, r)$*. Then, for any* $\boldsymbol{\phi} \in \boldsymbol{\Phi}$*, any* $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$*, and all polynomial time adversaries* $\mathcal{A}$*, we have*

$$\mathbf{Adv}^{(\boldsymbol{\phi},\boldsymbol{\phi}')\text{-cis}}_{\mathcal{H},\mathcal{A}}(\lambda) < \epsilon.$$

We will sketch the proof of the above theorem.

Proof (Sketch). If an adversary $\mathcal{A}$ has advantage greater than or equal to ϵ, we would be able to build an extractor adversary $\mathcal{D}$ that distinguishes the outputs of the extractor with probability ϵ. This in turn would allow us to build a function $\mathtt{Rec}$ that recovers r with probability greater than δ (cf. Definition 6), which is not possible by assumption. Hence, we have a contradiction, so the advantage of the adversary $\mathcal{A}$ must be less than ϵ. □

A concrete instantiation of such a CIS hash is possible via Theorem 1 of [10]. If we define

$$h_c(r) := \langle r, c \rangle, \tag{2}$$

where $c \in \mathbb{Z}_p^\lambda$ and $r \in H^\lambda$ for $H \subset \mathbb{Z}_p$, then the following corollary is obvious.

Corollary 1. *Consider the hash function family* $\mathcal{H}$ *defined by Equation 2, and let* $\boldsymbol{\Phi}'$ *be a set of* δ*-hard-to-invert functions. Then, for all* $\boldsymbol{\phi}' \in \boldsymbol{\Phi}'$*, and all polynomial time adversaries* $\mathcal{A}$*, we have*

$$\mathbf{Adv}^{(\mathrm{id},\boldsymbol{\phi}')\text{-cis}}_{\mathcal{H},\mathcal{A}}(\lambda) < \sqrt[3]{512\delta\lambda p^3}.$$

As highlighted above, CIS hash functions share interesting connections with RKA-secure primitives. In fact, [14] proposed a general approach for obtaining RKA-security via a CIS hash function. For example, consider a standard signature scheme given by algorithms $\{\mathtt{KeyGen}, \mathtt{Sign}, \mathtt{Verify}\}$, and a $(\mathrm{id}, \boldsymbol{\Phi}')$-pseudorandom correlated-input secure hash function h for which we assume a key c is publicly available. To obtain a RKA-secure signature scheme for functions $\boldsymbol{\Phi}'$, simply replace the random coins r used by $\mathtt{KeyGen}$ with $h_c(r)$, and the signing key with r. Furthermore, since the signing key of the original scheme is no longer stored, the algorithm $\mathtt{Sign}$ must regenerate this from r using h_c and $\mathtt{KeyGen}$. As shown in [14], the resulting signature scheme will be RKA-secure for functions $\boldsymbol{\Phi}'$.

Note that, in this approach, only a $(\mathrm{id}, \boldsymbol{\Phi}')$-pseudorandom correlated-input secure hash function is required. Hence, by using the CIS hash function from Corollary 1 in the above sketched transformation, we can obtain a RKA-secure signature scheme for hard-to-invert functions. As far as the authors are aware, this is the first construction of a RKA-secure signature scheme for this class of functions. Furthermore, a similar result can be obtained for any primitive for

which the above transformation applies. However, note that due to the properties of the above described security model for CIS hash functions, which implies that the functions $\boldsymbol{\Phi}'$ are independent of the hash function key, we only obtain selective RKA-security.

References

1. Michel Abdalla, Fabrice Benhamouda, Alain Passelègue, and Kenneth G. Paterson. Related-key security for pseudorandom functions beyond the linear barrier. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO (1)*, volume 8616 of *Lecture Notes in Computer Science*, pages 77–94. Springer, 2014.
2. Andrew Becherer, Alex Stamos, and Nathan Wilcox. Cloud computing security: Raining on the trendy new parade. *BlackHat USA*, 2009.
3. Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Hedged public-key encryption: How to protect against bad randomness. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 2009.
4. Mihir Bellare and David Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 666–684. Springer, 2010.
5. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.
6. Mike Bendel. Hackers describe PS3 security as epic fail, gain unrestricted access, 2011. `http://www.exophase.com/20540/hackers-describe-ps3-security-as-epic-fail-gain-unrestricted-access/`.
7. Daniel J. Bernstein, Yun-An Chang, Chen-Mou Cheng, Li-Ping Chou, Nadia Heninger, Tanja Lange, and Nicko van Someren. Factoring RSA keys from certified smart cards: Coppersmith in the wild. Cryptology ePrint Archive, Report 2013/599, 2013. `http://eprint.iacr.org/`.
8. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.
9. Debian. Debian Security Advisory DSA-1571-1: OpenSSL – predictable random number generator, 2008. `http://www.debian.org/security/2008/dsa-1571`.
10. Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 361–381. Springer, 2010.
11. Yevgeniy Dodis, David Pointcheval, Sylvain Ruhault, Damien Vergnaud, and Daniel Wichs. Security analysis of pseudo-random number generators with input: /dev/random is not robust. *IACR Cryptology ePrint Archive*, 2013:338, 2013.
12. Leo Dorrendorf, Zvi Gutterman, and Benny Pinkas. Cryptanalysis of the random number generator of the Windows operating system. *ACM Trans. Inf. Syst. Secur.*, 13(1), 2009.
13. Ian Goldberg and David Wagner. Randomness and the Netscape browser, 1996. `http://www.drdobbs.com/windows/184409807`.

14. Vipul Goyal, Adam O'Neill, and Vanishree Rao. Correlated-input secure hash functions. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2011.
15. Zvi Gutterman and Dahlia Malkhi. Hold your sessions: An attack on java session-id generation. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 44–57. Springer, 2005.
16. Zvi Gutterman, Benny Pinkas, and Tzachy Reinman. Analysis of the linux random number generator. In *IEEE Symposium on Security and Privacy*, pages 371–385. IEEE Computer Society, 2006.
17. Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *USENIX Security Symposium*, August 2012.
18. Kaoru Kurosawa and Yvo Desmedt. A new paradigm of hybrid encryption scheme. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2004.
19. Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter. Public keys. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 626–642. Springer, 2012.
20. Stefan Lucks. Ciphers secure against related-key attacks. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 359–370. Springer, 2004.
21. Kai Michaelis, Christopher Meyer, and Jörg Schwenk. Randomly failed! the state of randomness in current java implementations. In Ed Dawson, editor, *CT-RSA*, volume 7779 of *Lecture Notes in Computer Science*, pages 129–144. Springer, 2013.
22. Kenneth G. Paterson, Jacob C. N. Schuldt, and Dale L. Sibborn. Related randomness attacks for public key encryption. In Hugo Krawczyk, editor, *PKC*, volume 8383 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2014.
23. Kenneth G. Paterson, Jacob C. N. Schuldt, and Dale L. Sibborn. Related randomness attacks for public key encryption. *IACR Cryptology ePrint Archive*, 2014:337, 2014.
24. Ananth Raghunathan, Gil Segev, and Salil P. Vadhan. Deterministic public-key encryption for adaptively chosen plaintext distributions. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 93–110. Springer, 2013.
25. Thomas Ristenpart and Scott Yilek. When good randomness goes bad: Virtual machine reset vulnerabilities and hedging deployed cryptography. In *NDSS*. The Internet Society, 2010.
26. Hoeteck Wee. Public key encryption against related key attacks. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2012.
27. Daniel Wichs. Barriers in cryptography with weak, correlated and leaky sources. In Robert D. Kleinberg, editor, *ITCS*, pages 111–126. ACM, 2013.
28. Scott Yilek. Resettable public-key encryption: How to encrypt on a virtual machine. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 41–56. Springer, 2010.

Setup
$(pk^*, sk^*) \leftarrow \texttt{PKE.K}(1^\lambda)$
$(\hat{pk}^*, \hat{sk}^*) \leftarrow (pk^* || s, sk^*)$
$b \leftarrow_\$ \{0,1\}$

LR query (m_0, m_1, i)
$r' \leftarrow F_{z_i}(\hat{pk}^* || m_b)$
$c \leftarrow \texttt{PKE.E}(pk^*, m_b; r')$
return c

Enc query $(\hat{pk}, m, i)$
return $\texttt{EXT-PKE.E}(\hat{pk}, m; \phi_i(r))$

Dec query c
return $\texttt{PKE.D}(sk^*, c)$

Fig. 7. Simulation used in the proof of Lemma 2

A Proof of Theorem 1

Proof. First, we invoke Lemma 1, so that we now only have to prove the theorem for an adversary using just one randomness value. Furthermore, we assume that an adversary never repeats a query. Identical queries will result in identical outputs, hence any adversary that repeats a query may be replaced by a more efficient adversary with the same advantage. We proceed via a sequence of game hops. The games are as follows:

G_0: G_0 is the real game with the scheme defined in Figure 4.
G_1: G_1 is the same as G_0, except for **LR** queries, where the output of the extractor is replaced with a uniformly random value. If there existed an adversary that could distinguish between these two games, then we could use this adversary to reconstruct $\phi_i(r)$.
G_2: G_2 is the same as G_1, except for **LR** queries, where the outputs of the PRF are replaced with uniformly random values. Finally, G_2 may be simulated by a standard IND-ATK adversary.

G_0 – G_1: We will prove the following:

Lemma 2. *For any adversary $\mathcal{A}$, the difference in success probabilities in games G_0 and G_1 is bounded by ϵ:*

$$\Pr[G_1^{\mathcal{A}} \Rightarrow 1] - \Pr[G_0^{\mathcal{A}} \Rightarrow 1] < \epsilon.$$

Proof. Consider a distinguisher $\mathcal{D}$, attempting to distinguish whether the extractor game has returned a real or random output. Adversary $\mathcal{D}$ is given a seed s, two vectors of functions $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_q)$ and $\boldsymbol{\phi}' = (\phi'_1, \ldots, \phi'_{q'})$, and the vector $\boldsymbol{\phi}' = (\phi'_1(r), \ldots, \phi'_q(r))$, and a value $\boldsymbol{z} = (z_1, \ldots, z_q)$ which is either $\{\texttt{Ext}(\phi_i(r); s)\}_{i \in \{1,\ldots,q\}}$ or $(\sigma_1, \ldots, \sigma_q)$ where each σ_i is chosen uniformly at random from the range Σ of the extractor. The distinguisher $\mathcal{D}$ sets up the simulation as shown in **Setup** in Figure 7, and forwards the public key $\hat{pk}^*$ to $\mathcal{A}$. Then $\mathcal{D}$ answers $\mathcal{A}$'s queries as shown in **Enc query**, **LR query**, and **Dec query** in Figure 7.

When $\mathcal{A}$ halts and outputs b', $\mathcal{D}$ halts and outputs 1 if and only if $b = b'$. If $\boldsymbol{z} = \{\texttt{Ext}(\phi_i(r); s)\}_{i \in \{1,\ldots,q\}}$ a perfect simulation of G_0 is provided. Otherwise, a perfect simulation of G_1 is provided. Hence,

$$|\Pr[G_0^{\mathcal{A}} \Rightarrow 1] - \Pr[G_1^{\mathcal{A}} \Rightarrow 1]| = |\Pr[\mathcal{D}(s, \texttt{Ext}(r; s), \boldsymbol{\phi}'(r)) \Rightarrow 1] - \Pr[\mathcal{D}(s, \sigma, \boldsymbol{\phi}(r)) \Rightarrow 1]|.$$

Setup
$b \leftarrow_\$ \{0,1\}$
$s \leftarrow_\$ \texttt{seeds}$
$r \leftarrow_\$ \texttt{Rnd}$
$K_{j+2}, \ldots, K_q \leftarrow_\$ \texttt{PRF.K}(1^\lambda)$
$(pk^*, sk^*) \leftarrow \texttt{PKE.K}(1^\lambda)$
$(\hat{pk}^*, \hat{sk}^*) \leftarrow (pk^* || s, sk^*)$

Enc query $(\hat{pk}, m, i)$
return $\texttt{EXT-PKE.E}(\hat{pk}, m; \phi_i(r))$

Dec query c
return $\texttt{PKE.D}(sk, c)$

LR query (m_0, m_1, i)
if $i < j+1$
$r' \leftarrow_\$ \texttt{Rnd}$
if $i = j+1$
forward $pk^* || m_b$ to PRF oracle;
receive output r'
if $i > j+1$
$r' \leftarrow F_{K_i}(pk^* || m_b)$
$c \leftarrow \texttt{PKE.E}(\hat{pk}^*, m_b; r')$
return c

Fig. 8. Simulation used in the proof of Lemma 3

If an adversary can distinguish outputs of the extractor from uniformly random values with probability greater than or equal to ϵ in polynomial time, then there exists an algorithm Rec that will recover $\phi_i(r)$ (for some $i \in \{1, \ldots, q\}$) with probability greater than δ (cf. Definition 6). However, this is a contradiction since the $\boldsymbol{\phi}$ is δ-hard-to-compute with respect to $\boldsymbol{\phi}'$. Hence, we must have

$$\Pr[G_1^{\mathcal{A}} \Rightarrow 1] - \Pr[G_0^{\mathcal{A}} \Rightarrow 1] < \epsilon.$$

□

$\mathbf{G_1}$ – $\mathbf{G_2}$: If an adversary can distinguish games 1 and 2 with a certain probability, then we may construct a PRF adversary $\mathcal{B}$ that wins the PRF game with the same probability. Specifically:

Lemma 3. *The difference between games G_1 and G_2 is bounded by the advantage of a PRF adversary. More specifically,*

$$|\Pr[G_1^{\mathcal{A}} \Rightarrow 1] - \Pr[G_2^{\mathcal{A}} \Rightarrow 1]| \leq \mathbf{Adv}^{\mathrm{prf}}_{F,\mathcal{B}}(\lambda).$$

Proof. The proof uses a hybrid argument. Consider the hybrid game $G_{1,j}$, where the outputs of the PRF for functions $\{\phi_i\}_{i<j+1}$ are replaced with uniformly random values. Notice that $G_1 = G_{1,0}$ and $G_2 = G_{1,q}$. $\mathcal{B}$ sets up the simulation of $G_{1,j}$ as shown in **Setup** in Figure 8, and returns the public key $\hat{pk}^*$ to $\mathcal{A}$. Then $\mathcal{B}$ answers $\mathcal{A}$'s queries as shown in **Enc query**, **LR query**, and **Dec query** in Figure 8.

When $\mathcal{A}$ halts and outputs b', $\mathcal{B}$ halts and outputs 1 if and only if $b = b'$. When $\mathcal{B}$'s outputs are from a PRF he simulates $G_{1,j}$ perfectly. Otherwise, if $\mathcal{B}$'s outputs are uniformly random he simulates $G_{1,j+1}$ perfectly. Hence, we have

$$|\Pr[G_1^{\mathcal{A}} \Rightarrow 1] - \Pr[G_2^{\mathcal{A}} \Rightarrow 1]| \leq q \cdot \mathbf{Adv}^{\mathrm{prf}}_{F,\mathcal{B}}(\lambda).$$

□

Setup
$r \leftarrow_{\$} \texttt{Rnd}$
$s \leftarrow_{\$} \texttt{seeds}$
$\text{ctr} \leftarrow 1$
$(pk^*, sk^*) \leftarrow \texttt{PKE.K}(1^\lambda)$
$(\hat{pk}^*, \hat{sk}^*) \leftarrow (pk^* || s, sk^*)$

Enc query (pk, m, i)
return $\texttt{EXT-PKE.E}(pk, m; \phi_i(r))$

Dec query c
return $\texttt{PKE.D}(sk, c)$

LR query (m_0, m_1)
If $\text{ctr} < j$,
$r' \leftarrow_{\$} \texttt{Rnd}$
$c \leftarrow \texttt{PKE.E}(pk, m_0; r')$
Else if $\text{ctr} > j$,
$r' \leftarrow_{\$} \texttt{Rnd}$
$c \leftarrow \texttt{PKE.E}(pk, m_1; r')$
Else submit (m_0, m_1) to $\mathcal{C}$'s **LR** oracle, receive c
$\text{ctr} \leftarrow \text{ctr} + 1$
return c to $\mathcal{A}$.

Fig. 9. Hybrid game $G_{2,j}$ used in the proof of Lemma 4

Finally, the game G_2 may be simulated by a standard IND-ATK PKE adversary, $\mathcal{C}$. More formally,

Lemma 4. *The success probability of an adversary attacking G_2 can be bound as follows:*

$$2 \cdot \Pr[G_2^{\mathcal{A}} \Rightarrow 1] - 1 \leq q_{LR} \cdot \mathbf{Adv}^{\text{ind-atk}}_{\texttt{PKE},\mathcal{C}}(\lambda).$$

Proof. This proof uses a hybrid argument. Specifically, consider a hybrid game $G_{2,j}$ in which the first j **LR** queries are answered with encryptions of m_0, and the following $q_{LR} - j$ LR queries are answered with encryptions of m_1. Specifically, game $G_{2,j}$ is set up as shown in **Setup** in Figure 9, and the adversaries queries are answered as shown in **Enc query**, **LR query**, and **Dec query** in Figure 9.

Note that the advantage of an adversary in game G_2 is equivalent to the adversary's ability to distinguish $G_{2,0}$ and $G_{2,q_{LR}}$. Furthermore, it is easily seen that, from an adversary $\mathcal{A}$ which is able to distinguish game $G_{2,j}$ and $G_{2,j+1}$, we can construct an IND-ATK adversary $\mathcal{C}$ against PKE with the same advantage, by using $\mathcal{C}$'s challenge as the response to the jth **LR** query. Hence, we conclude that

$$2 \cdot \Pr[G_2^{\mathcal{A}} \Rightarrow 1] - 1 \leq q_{LR} \cdot \mathbf{Adv}^{\text{ind-atk}}_{\texttt{PKE},\mathcal{C}}(\lambda).$$

□

The theorem follows by combining all of these inequalities. □