Correlated-Input Secure Hash Functions

Vipul Goyal[1], Adam O'Neill[2⋆], and Vanishree Rao[3⋆]

[1] Microsoft Research, India, vipul@microsoft.com
[2] University of Texas at Austin, adamo@cs.utexas.edu
[3] University of California, Los Angeles, vanishri@cs.ucla.edu

Abstract. We undertake a general study of hash functions secure under *correlated inputs*, meaning that security should be maintained when the adversary sees hash values of many related high-entropy inputs. Such a property is satisfied by a random oracle, and its importance is illustrated by study of the "avalanche effect," a well-known heuristic in cryptographic hash function design. One can interpret "security" in different ways: e.g., asking for one-wayness or that the hash values look uniformly and independently random; the latter case can be seen as a generalization of correlation-robustness introduced by Ishai et al. (CRYPTO 2003). We give specific applications of these notions to password-based login and efficient search on encrypted data. Our main construction achieves them (without random oracles) for inputs related by *polynomials* over the input space (namely $\mathbb{Z}_p$), based on corresponding variants of the q-Diffie Hellman Inversion assumption. Additionally, we show relations between correlated-input secure hash functions and cryptographic primitives secure under related-key attacks. Using our techniques, we are also able to obtain a host of new results for such related-key attack secure cryptographic primitives.

1 Introduction

In practice is often useful to view a cryptographic hash function like a random oracle, as formalized in the random oracle model [6]. However, as random oracles do not exist in reality (and indeed, in general the random oracle model may lead to insecure schemes [13]), an important line of research suggested by [13] seeks to formalize various useful properties satisfied by a random oracle and construct hash functions meeting them under standard assumptions. In this paper, we do so for what we call *correlated-input security*, meaning that (various notions of) security should be maintained when the adversary sees hash values of many related high-entropy inputs.

The importance of correlated-input security in practice is illustrated by the so-called *avalanche effect*, a well-known heuristic in cryptographic hash function design. (The name "avalanche effect" was coined by Feistel [17], although the idea goes back to Shannon's notion of diffusion [26].) Roughly, the avalanche effect states that making any change to an input should result in a drastically

⋆ Work done in part while at Microsoft Research, India.

different hash value. Clearly, such a hash function should satisfy a notion of correlated-input security. Our results help to shed light on whether or not this is feasible from a theoretical perspective.

1.1 Notions of Correlated-Input Security

We get different specific notions of correlated-input security depending on how we interpret "security." We first discuss the different interpretations we consider and and how we formalize the resulting notions.

Three Notions The first and most basic interpretation we consider is "one-wayness." To formalize one-wayness under correlated-inputs, we consider a hash function H and circuits $C_1, \ldots, C_n$, where each C_i takes as input some random coins and outputs a point in the domain of H. The adversary is given hash values $H(x_1), \ldots, H(x_n)$ where each x_i is the output of $C_i(r)$ for random coins r. Note that each C_i is run on the *same* random coins. Therefore the x_i are correlated.[4] The adversary's goal is to output any one of the x_i. Informally, we say that H is one-way under correlated inputs for a class of circuits $\{C\}$ if for any n and any choice of $C_1, \ldots, C_n$ from $\{C\}$, any efficient adversary succeeds with at most negligible probability.

The next interpretation we consider is "unpredictability." To formalize unpredictabililty under correlated-inputs, we consider a hash function H and circuits $C_1, \ldots, C_{n+1}$, where each C_i is as before. Now the adversary is given hash values $H(x_1), \ldots, H(x_n)$ and tries to output $H(x_{n+1})$, where each x_i is the output of $C_i(r)$ as before. The notion is defined for a class of circuits $\{C\}$ analogously to the one-wayness case. It mainly serves as a stepping-stone to our final notion.

Finally, the last interpretation we consider is "pseudorandomness." To formalize pseudorandomness under correlated-inputs, we consider a hash function H and circuits $C_1, \ldots, C_{n+1}$, each C_i is as before. Now the adversary is given hash values $H(x_1), \ldots, H(x_n)$ as well as a "challenge" value that is either $H(x_{n+1})$ or a random string of appropriate length, where each x_i is the output of $C_i(r)$ as before. (This of course requires the circuits to have distinct outputs.) Again, the notion is defined for a class of circuits $\{C\}$ analogously.

Discussion We make a few observations about these notions. One is that they are only achievable for a class of circuits $\{C\}$ such that $C(r)$ for random r has sufficient min-entropy for any C in the class. In fact, it is not hard to show that a random oracle satisfies our notions for the class of all such circuits. However, in the standard model they are in general only achievable by a *keyed* hash function H. To see this, fix an unkeyed hash function H and consider circuits C_1, C_2 where $C_1(r)$ outputs r and $C_1(r)$ outputs $H(r)$. Clearly, no fixed H is even one-way under correlated-inputs for these circuits. By a similar argument, the circuits must not depend on the choice of the hash key. We stress that in our notions the hash function key is *public*. Similar counter-examples show that in general

[4] For example, the x_i's might agree in most bit positions but vary in the others. It may even be the case that a single input x_i completely determines the rest.

pseudorandom generators and functions do not meet our notions (note that the latter has a *secret* key); we give details in the full version.

Another point is that when considering non-uniform inputs, these notions are non-trivial to achieve even in the case of a single input. However, this can be done; for example [27] considers one-way functions for a non-uniform input and [16] considers pseudorandom generators (that expand the input, which we do not require) for non-uniform seed. Additionally, we note for any *a priori bounded* number of circuits, pseudorandomness under correlated-input can be met even under information-theoretic indistinguishability (where the key-size depends on the bound). This follows from a generalization of the Leftover Hash Lemma [24, Lemma 6.1] (which follows [22, Lemma 3.2]). On the other hand, the focus of our work is on *correlations* among an *unbounded* number of inputs.

1.2 Applications

We now discuss some specific practical applications for our new notions.

Password-based login An application of our notion of one-wayness under correlated-inputs is password-based login. For example, UNIX maintains a "password" file that, for each user in the system, stores a hash of their password that is compared against the hash of a candidate password supplied at login by someone claiming to be this user. The goal is to prevent an adversary with access to the password file from gaining the ability to impersonate a user. Informally, it is often said that the property of the hash function needed to ensure this is one-wayness. But the standard notion of one-wayness is obviously insufficient here. Passwords, while they should contain entropy, are certainly not uniformly random. Moreover, passwords are typically *correlated*, both across different users, and across the same user on different systems (and the adversary may recover the password file for multiple systems).

This issue seems to be largely ignored in prior work. A paper (which we already mentioned) that considers the relevance of one-way functions for high entropy inputs to this application is [27]; however, they do not consider multiple related inputs and relations among them. Our notion of one-wayness under correlated input seems to be an appropriate security notion for this application.[5]

Efficient search on encrypted data An application of our notion of pseudorandomness under correlated-inputs is efficient search on encrypted data. It is becoming increasingly common for companies to store large amounts of data remotely on servers maintained by an untrusted third party. To provide privacy for

[5] For simplicity, this ignores "salting" the passwords, which can be viewed as considering a randomized hash function. This may make the problem easier for an approach based on the Leftover Hash Lemma (using a similar argument to [24, Lemma 6.1]), but as discussed in [27] such an approach is impractical due to its "entropy loss." For our approach it does not make the problem significantly easier, and anyway it circumvents the core issue that in practice a (deterministic) cryptographic hash function is assumed to satisfy correlated-input security; indeed, salting passwords is only meant to slow down dictionary attacks.

the client, the data should be encrypted. However, we still want to allow search on the data without retrieving and decrypting the entire database. Techniques like public-key encryption with keyword search [11] make search possible, but it takes linear time in the database size. On the other hand, practitioners require search time to be comparable to that for unencrypted data.

This problem was first studied from a cryptographic perspective by Bellare et al. [2], who introduced deterministic encryption and the more general concept of efficiently searchable encryption (ESE) as a solution. The basic idea is to attach a hash of each keyword to an encrypted file. Keywords are obviously not uncorrelated, and thus our notions are natural to apply. In fact, if a hash function meets our notion of pseudorandomness under correlated-inputs, then it can be "bootstrapped" to hide all partial information information by encrypting a hash value under the one-way trapdoor permutation based deterministic encryption scheme of [4] (actually, a one-way permutation without a trapdoor suffices here, since we do not need to decrypt).

1.3 Our Construction and its Security

Next we turn to whether our security definitions can be achieved and under what cryptographic assumptions.

Our Construction We propose the following construction: Letting G be a group of prime order p, the hash key is a random generator $g \in G$ and random $c \in \mathbb{Z}_p$, and the evaluation of the hash on input $x \in \mathbb{Z}_p$ is $g^{1/(x+c)}$ where $1/(x+c)$ denotes the inverse of $x+c$ modulo p.

Security Analysis We show that this construction is secure under each of our three notions of security assuming (appropriate variants of) the *q-Diffie-Hellman inversion assumption* (q-DHI). Roughly speaking, this assumption says that given $g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^q}$, it is hard to compute $g^{1/\alpha}$. An assumption of this form was first introduced by Boneh and Boyen [9], who considered it in groups with a *a bilinear map* (pairing) e and asked that it be hard to compute (or distinguish from random) $e(g,g)^{1/x}$ instead of $g^{1/x}$. However, since our hash function is deterministic and thus automatically publically verifiable, we do not need bilinear maps here.

The class of circuits we consider in our proofs are the ones that are (efficiently) representable by a polynomial over $\mathbb{Z}_p$ (the input space of the hash function). In other words, each $x_i = p_i(x, y, z, \ldots)$ where p_i is a polynomial over $\mathbb{Z}_p$ the $x, y, z, \ldots$ are randomly chosen but fixed for all i. In our main theorems, we treat the case of univariate polynomials p_i over $\mathbb{Z}_p$, but it is easy to extend our proofs to multivariate polynomials as well. This is quite a broad class, in particular just considering univariate polynomials and taking p_1 to be the identity polynomial covers the well-known attacks on RSA [15].

First, we show that our inversion hash is one-way under correlated inputs for this class of circuits assuming the q-strong discrete logarithm (q-SDL) assumption (which is weaker than q-DHI in that it need only be hard to compute

α itself).[6] For unpredictability and psuedorandomness, we require an additional assumption that each input to the hash function is individually uniform.[7] However we stress that given other inputs, an input may even be fully computable. For this class of polynomials, we show the inversion hash is unpredictable under correlated inputs assuming q-DHI. Using standard hardcore bit techniques this already gives a construction with small output length achieving pseudorandomness under correlated inputs with the same assumption. However, we directly show pseudorandomness under correlated inputs of our construction for the same class of polynomials assuming the decisional version of q-DHI.

Discussion One may notice the similarity of our construction to the Boneh-Boyen short signature scheme [10]. Our proof techniques build on those introduced in [10], but note that, as opposed to the latter, we focus on a setting where there is no secret key, and we use the q-DHI assumption instead of the q-SDH assumption of [10]. Indeed, our proofs use some new ideas. In particular, the role of c in our reductions for unpredictability and pseudorandomness is completely different from that of the message in [10].

We also note that our security proofs are under a notion of "selective" security where the circuits that sample the inputs do not depend on the public hash key. As we mentioned, in general this restriction is inherent. However, for restricted classes of circuits (such as arithmetic circuits we consider) which are not able to efficiently compute the hash function in question, it may be possible to achieve an adaptive notion of security even by using an *unkeyed* hash function. We discuss a positive result for this case below.

Finally, we note that our construction as defined is not compressing. However, once we obtain a construction meeting any of our notions, it is easy to obtain one which is also compressing. In the case of one-wayness we can apply a collision-resistant hash to the output, and in the case of pseudorandomness we can truncate the output. It can be shown that the resulting (compressing) hash function retains correlated-input security.

1.4 Relations to Related-Key Attacks

Security under related-key attacks (RKA), first formalized by [5] in the context of pseudorandom functions/permutations, is a well-established notion that, like correlated-input security, asks for security to be maintained under related values of a "secret" input. We explore elations between pseudorandomness under correlated-inputs secure hash (which we simply call CI-secure hash below) and RKA-security of various cryptographic primitives.

[6] In fact, for this result the c component of the hash key can be any fixed element in $\mathbb{Z}_p$ (for instance, set $c = 0$).

[7] This translates to the requirement that the polynomials individually have uniform output on uniform input (e.g., this is the case for permutation polynomials). By making non-standard assumptions, it may be possible to drop this restriction. However, considering individually uniform but correlated inputs to the hash function is natural, and we focus on results under standard assumptions.

Equivalence to One-Input RKA-Scure wPRF We first observe that a CI-secure hash is in some sense equivalent to what we call a "one-input RKA-secure weak pseudorandom function (wPRF)." To define such a wPRF F, the adversary is given an input-output pair $x, F(K, x)$ for random x and may query for for other outputs of the form $F(\phi(K), x)$ for relations ϕ of its choosing. (Note that the same x is re-used each time.) Following [20], we note that as compared to a CI-secure hash, the role of the key and the input are simply "switched."[8] Note that a one-input RKA-secure wPRF is implied by an RKA-secure PRF; in fact, if we start with an RKA-secure PRF (rather than wPRF), the resulting CI-secure hash *does not need a public key*. The latter is significant because [3] gives RKA-secure PRFs, in particular one under the decisional Diffie-Hellman assumption for adaptively chosen, group-induced relations (i.e., multiplication by a constant). We thus obtain an unkeyed adaptively-secure CI-secure hash for the corresponding class of circuits.

A General Transformation for RKA Security Addtionally, we propose a transformation to "bootstrap" any cryptographic primitive to one that is RKA-secure: simply hash the coins used to generate the secret key for the former. (This can be seen as replacing a RO in this transformation with a CI-secure hash.) Note, however, that in the case the CI-secure hash has a public key, an authentic version of the latter is then needed by any algorithm that uses the secret key (e.g., the signing algorithm for a signature scheme), which may not always be practical. Additionally, while our main construction of CI-secure hash is selectively secure, leading to selective security under RKA (i.e., relations chosen before seeing the public parameters). However, by using our techniques in a non-blackbox way, we can sometimes achieve adaptive security instead and without any public key. In particular, we show how to do this for RKA-secure symmetric encryption, a primitive introduced concurrently to our work in [1];[9]. In this case, we are even able to handle the entire class of (non-zero) polynomial relations.

1.5 Other Related Work

Our work is related to several other lines of research, as we now discuss.

Realizing Random Oracles As we mentioned, our work can be seen as extending a research agenda proposed by Canetti, Halevi, and Goldriech [13], in which one identifies and realizes useful properties of a random oracle (RO). Indeed, by using the techniques of [2, Theorem 5.1] one can show that a RO meets all the security definitions we consider (which is why we have sought realizations under standard cryptographic assumptions). Other useful properties of a RO that have undergone a similar treatment include perfect one-wayness [12, 14] and non-malleability [7] We note that none of these works address security under multiple correlated inputs.

[8] Note, however, that CI-input security is more general than RKA-security in that it considers inputs sampled from a common "history."

[9] We obtained this result after seeing [1], but the rest of our work was concurrent.

In fact, a significant prior work of Ishai et al. [21, Definition 1] considered a notion of "correlation-robustness" for pseudorandom hash functions, with the motivation of instantiatng the RO in their oblivious transfer protocols. Their notion is more restrictive than our notion of pseudorandomness under correlated-input, as the former is defined only for hash functions that output a single bit and considers inputs obtained by computing the exclusive-or's of a "master" random input s with public random values.

Finally, we note that while realizing the "avalanche effect" satisfied by a RO forms a major motivation for considering correlated-input security, it is not the only way the latter could be formalized. In particular, it talks only about the change in the output behavior relative to any change to an *unknown* input. The notion of "(multiple input) correlation intractability" due to [13] is a possible formalization the effect without this restriction. On the other hand, the latter notion seems harder to work with and more difficult to achieve.

Deterministic Encryption Our security notions can be viewed as relaxations or variants of the notions of privacy proposed for deterministic encryption (DE) in [2, 8, 4, 24] (that seek to hide partial information) in the case of hash functions rather than encryption schemes. Indeed, the results of of [2, 8, 4, 24] show that in some sense the "hard part" of realizing DE without random oracles is dealing with correlations among the inputs. Our work studies this issue at a more basic level, asking whether it is feasible even without supporting decryption and for weaker security notions like one-wayness.

Related Security Notions Recently, Rosen and Segev [25] introduced the notion of *correlated-product secure* trapdoor functions (TDFs). Correlated-product security was later considered for hash functions in [20]. Correlated-product security differs from our notions in that the former refers to security when related inputs are evaluated under *independent instances* of the function; in other words, there does not exist a single function which is evaluated on related inputs (as is the case for correlated-input security). Indeed, our techniques are quite different and unrelated to those in [25, 20].

The recent work of Goldenberg and Liskov [19] also considers a form of correlated-input security (which they call "related-secret" security) for various primitives. While their work has some similarities to ours (for example, they consider "related-secret" one-way functions), there are some important differences. Namely, they focus on hardcore bits and pseudorandom functions rather than hash functions, and they follow the definitional framework for RKA-security introduced in [5]. As mentioned above, this definitional framework is less general than ours. Additionally, their results are mainly negative.

2 Preliminaries

Notations. Let $x \xleftarrow{\$} X$ denote the operation of selecting a random element x from X. Let by $x \leftarrow y$ denote the assignment of a value y to x. Let $|X|$ denote the size of the set X, and $|x|$ denote the length of the string x. Let λ denote the security parameter. Let $\mathsf{FF}(\mathcal{K}, \mathcal{D}, \mathcal{R})$ be the set of all families of functions

$F : \mathcal{K} \times \mathcal{D} \longrightarrow \mathcal{R}$. For sets X, Y, let $\mathsf{Fun}(X, Y)$ be the set of all functions mapping X to Y. For brevity, we say that an algorithm outputs a set/function as a shorthand to mean that it outputs their descriptions.

Complexity Assumptions. We first state our complexity assumptions, namely q-DHI [9, 10], which is weaker than q-BDHI [9] as well as q-SDH [10], and we introduce what we call the q-Strong Discrete Logarithm (q-SDL) assumption which is weaker than all of q-DHI, q-BDHI, and q-SDH assumptions.

Let GrpGen be a PPT algorithm that takes as input the security parameter 1^λ and outputs parameters for some cyclic multiplicative group $\mathbb{G}$, including the group order p which is a $poly(\lambda)$-bit integer, a generator g, and an efficient algorithm (e.g., circuit) for multiplication (and thus also exponentiation). We denote it as $(\mathbb{G}, p, g) \leftarrow \mathsf{GrpGen}(1^\lambda)$.

q-Strong Discrete Logarithm (q-SDL) Problem The q-SDL problem in $\mathbb{G}$ is defined as follows: given a $(q+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^q}) \in (\mathbb{G}^\star)^{q+1}$ for some unknown $x \in \mathbb{Z}_p^\star$, output x.
An algorithm $\mathcal{A}$ solves the q-SDL problem in the group $\mathbb{G}$ with advantage ϵ if

$$\mathsf{SDL}\ \mathsf{Adv}_{\mathcal{A},q} := \Pr[\mathcal{A}(g, g^x, g^{x^2}, \ldots, g^{x^q}) = x] \geq \epsilon$$

where the probability is over the random choice of generator $g \in \mathbb{G}^\star$, the random choice of $x \in \mathbb{Z}_p^\star$, and the random bits consumed by $\mathcal{A}$.

Definition 1. *We say that the (q, t, ϵ)-SDL assumption holds in $\mathbb{G}$ (or GrpGen satisfies the (q, t, ϵ)-SDL assumption) if no probabilistic t-time algorithm has advantage at least ϵ in solving the q-SDL problem in $\mathbb{G}$.*

It is easy to see that the 1-SDL assumption is equivalent to the standard Discrete Logarithm assumption.

q-Diffie-Hellman Inversion (q-DHI) Problem The q-DHI problem [9, 10] in $\mathbb{G}$ is defined as follows: given a $(q+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^q}) \in (\mathbb{G}^\star)^{q+1}$ for some unknown $x \in \mathbb{Z}_p^\star$, output $g^{\frac{1}{x}} \in \mathbb{G}$.
An algorithm $\mathcal{A}$ solves the q-DHI problem in the group $\mathbb{G}$ with advantage ϵ if

$$\mathsf{DHI}\ \mathsf{Adv}_{\mathcal{A},q} := \Pr[\mathcal{A}(g, g^x, g^{x^2}, \ldots, g^{x^q}) = g^{\frac{1}{x}}] \geq \epsilon$$

where the probability is over the random choice of generator $g \in \mathbb{G}^\star$, the random choice of $x \in \mathbb{Z}_p^\star$, and the random bits consumed by $\mathcal{A}$.

Definition 2. *We say that the (q, t, ϵ)-DHI assumption holds in $\mathbb{G}$ (or GrpGen satisfies the (q, t, ϵ)-DHI assumption) if no probabilistic t-time algorithm has advantage at least ϵ in solving the q-DHI problem in $\mathbb{G}$.*

q-DHI Problem can be equivalently stated as follows [10]: given a $(q+2)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^q}, c) \in (\mathbb{G}^\star)^{q+1} \times \mathbb{Z}_p \backslash \{x\}$ for some unknown $x \in \mathbb{Z}_p^\star$, output $g^{\frac{1}{x+c}} \in \mathbb{G}$. This definition also clearly points out the distinction between the q-DHI problem and the q-SDH problem: in case of the q-DHI problem, the value of c is prescribed in the problem instance itself, whereas in case of the q-SDH problem, the solver is free to choose $c \in \mathbb{Z}_p$ and output a pair, $(c, g^{\frac{1}{x+c}})$. Obviously, the q-DHI assumption is weaker than the q-SDH assumption.

Decisional q-Diffie-Hellman Inversion (Decisional q-DHI) Problem The decisional q-DHI problem in $\mathbb{G}$ is defined as follows: given a $(q+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^q}) \in (\mathbb{G}^\star)^{q+1}$ for some unknown $x \in \mathbb{Z}_p^\star$, distinguish between $g^{\frac{1}{x}}$ and a random element $R \xleftarrow{\$} \mathbb{G}$.
An algorithm $\mathcal{A}$ solves the decisional q-DHI problem in $\mathbb{G}$ with advantage ϵ if

$$\begin{aligned}\mathsf{DDHI}\ \mathsf{Adv}_{\mathcal{A},q} := |\Pr[\mathcal{A}(g, g^x, g^{x^2}, \ldots, g^{x^q}, g^{\frac{1}{x}}) = 1] \\ - \Pr[\mathcal{A}(g, g^x, g^{x^2}, \ldots, g^{x^q}, R) = 1]| \geq \epsilon\end{aligned}$$

where the probability is over the random choice of generator $g \in \mathbb{G}^\star$, the random choice of $x \in \mathbb{Z}_p^\star$, the random choice of $R \in \mathbb{G}^\star$, and the random bits consumed by $\mathcal{A}$. The distribution on the left is referred to as $\mathcal{P}_{\mathcal{DDHI}}$ and the distribution on the right as $\mathcal{R}_{\mathcal{DDHI}}$.

Definition 3. *We say that the decisional (q, t, ϵ)-DHI assumption holds in $\mathbb{G}$ (or GrpGen satisfies the decisional (q, t, ϵ)-DHI assumption) if no probabilistic t-time algorithm has advantage at ϵ in solving the decisional q-DHI problem in $\mathbb{G}$.*

3 Our Model: Correlated Input Security

In this section, we define our new notion of security for cryptographic hash functions. We would be interested in preserving various properties of hash functions (like one-wayness and pseudo-randomness) when the function maybe evaluated on a tuple of inputs which maybe be *correlated* in an arbitrary way. Standard notions of security do not provide any guarantee in such a setting. In the full version, we discuss examples of functions which are secure in the standard sense but may be completely insecure when evaluated on multiple inputs which are correlated.

Before we go further, we first discuss how we represent correlations among a tuple of inputs $(m_1, \ldots, m_n)$. In general, such an input tuple maybe generated by a polynomial-size sampler circuit *Samp*. In other words, *Samp* takes a random tape r (of appropriate length) as input such that $(m_1, \ldots, m_n) \leftarrow Samp(r)$. Note such a sampler circuit can generate the input tuple for any type of polynomial-time computable correlations. Equivalently, one can generate the (correlated) tuple of inputs using a *tuple* of polynomial-size circuits $(C_1, \ldots, C_n)$ when initialized on the same random tape. In other words, fix a random string r and set $m_i \leftarrow C_i(r)$. It is easy to see that both these sampling procedures are equivalent. For the rest of the paper, we shall stick to the latter mode of using a tuple of circuits (for convenience, as will be clear later on).

Also, it will be understood that the range of every circuit considered is a subset of the input-space (or keyspace) in question. We refer to an adversary as $\{C\}$-restricted, if its correlated-input circuits are restricted to the class of circuits $\{C\}$.

We first define the syntax for a general function family (or a hash function family if the functions are compressing). We will then move on to formalize the various security properties such a function family might satisfy.

Definition 4 (Function Family). *A family of deterministic functions $\mathcal{H}$ is specified by a PPT algorithm Gen. The algorithm Gen, given input 1^λ, outputs a parameter set $\mathcal{I}_h$, domain $\mathcal{D}_h$, and range $\mathcal{R}_h$, and outputs $c \in \mathcal{I}_h$ as a description of a function $h_c : \mathcal{D}_h \longrightarrow \mathcal{R}_h$. The sizes of the domain and range sets are each exponential in the security parameter.*

Now, we shall discuss our first notion of security called *correlated-input one-wayness*. Informally, we consider a function $h(\cdot)$ such that given $(h(m_1), \ldots, h(m_n))$, where inputs $(m_1, \ldots, m_n)$ maybe correlated, it is hard for any PPT adversary to output any valid preimage m_i. This can be viewed as a generalization of the standard notion of one-way functions. We allow the adversary to specify the correlations to the challenger by giving a tuple of circuits $(C_1, \ldots, C_n)$, where each circuit is from a class of correlated-input circuits $\{C\}$. Note that, for this definition to be satisfiable, each circuit $C_i, \in \{C\}$ individually should have high min-entropy output distribution for uniform random input distribution. [10] Thus, we quantify only over such circuits in our definition. We discuss it under both selective and adaptive security notions. More details follow.

In the following we shall only formalize the selective notion, while we refer the reader to the full version for definitions of the adaptive notions.

The Selective Correlated-Input Inverting experiment $\mathsf{Exp}^{sCI-inv}_{\mathcal{A},\mathcal{H},\{C\}}$. For a family of deterministic functions $\mathcal{H}$, an adversary $\mathcal{A}$, and a family of efficiently-computable correlated-input circuits $\{C\}$, we define the following game between a challenger and the adversary $\mathcal{A}$.

- **Setup Phase 1**. Challenger runs the Gen algorithm of $\mathcal{H}$ for a security parameter input 1^λ and gets $h_c : \mathcal{D}_h \longrightarrow \mathcal{R}_h$. Challenger gives $\mathcal{D}_h$ to $\mathcal{A}$.
- **Query Phase**. $\mathcal{A}$ chooses a positive integer n $(= poly(\lambda))$, and gives to the challenger n circuits $\{C_i\}_{i\in[n]} \subset \{C\}$.
- **Setup Phase 2**. Challenger gives $h_c(\cdot)$ to $\mathcal{A}$ and chooses r, a uniform random string of appropriate length.
- **Response Phase**. $\forall i \in [n]$, challenger responds via $h_c(C_i(r))$.
- **Invert Phase**. $\mathcal{A}$ outputs $(\hat{k}, \hat{y})$ for $\hat{k} \in [n]$ and $\hat{y} \in \mathcal{D}_h$.

The output of the experiment is defined to be 1 if $h_c(\hat{y}) = h_c(C_{\hat{k}}(r))$ and 0 otherwise.

We define the advantage of an adversary $\mathcal{A}$ in the above game as:

$$\mathsf{Adv}^{sCI-inv}_{\mathcal{A},\mathcal{H},\{C\}}(\lambda) = \Pr[\mathsf{Exp}^{sCI-inv}_{\mathcal{A},\mathcal{H},\{C\}} = 1]$$

The probability is over the random bits used by the challenger and the adversary.

[10] This requirement is similar to one in the standard notion of one-way functions. If the input does not have sufficient min-entropy, it is easy to see that an adversary can guess a preimage and succeed with noticeable probability.

Definition 5. *A family of functions $\mathcal{H}$ is said to be selective correlated-input one-way with respect to a family of correlated-input circuits $\{C\}$, if for all $\mathcal{A} \in$ PPT there exists a negligible function negl, such that:*

$$Adv^{sCI-inv}_{\mathcal{A},\mathcal{H},\{C\}}(\lambda) \leq negl(\lambda)$$

We now consider two more correlated-input security notions where we talk about the unpredictability of the *output* as opposed to that of the input. Informally, we consider a function $h_c : \mathcal{D}_h \longrightarrow \mathcal{R}_h$ with the following properties. Consider a tuple of correlated inputs $(m_1, \ldots, m_{n+1})$. The adversary is given the function outputs $(h_c(m_1), \ldots, h_c(m_n))$ and it tries to compute $h_c(m_{n+1})$. In the first security notion called *correlated-input unpredictability* (CI-unpredictability), we require that it should be hard for the adversary to output $h_c(m_{n+1})$. In the next notion called *correlated-input pseudorandomness* (CI-pseudorandomness), we require that the adversary should not be able to distinguish $h_c(m_{n+1})$ from a random element in $\mathcal{R}_h$, given $(h_c(m_1), \ldots, h_c(m_n))$. It is easy to show that this notion of CI-pseudorandomness is equivalent to a notion where an adversary gets either $(h_c(m_1), \ldots, h_c(m_{n+1}))$ or $n+1$ independent random elements in $\mathcal{R}_h$ and is required to distinguish the two cases. Note that, for any of these notions to be satisfiable, besides the requirement that each circuit $C_i \in \{C\}$ individually should have high min-entropy output distribution for uniform random input distribution, we also require that, for every two distinct circuits C_i and C_j in $\{C\}$, and for a uniform random input r of appropriate length, $C_i(r) = C_j(r)$ happens only with negligible probability over the choice of r.

In trying to give more power to the adversary (thus making our definition stronger), we allow the adversary to specify the correlation by giving a tuple of circuits $(C_1, \ldots, C_{n+1})$, where $C_i \in \{C\}$, to the challenger, as before. In addition, for the selective case, for a tuple of inputs $(m_1, \ldots, m_{n+1})$, we allow the adversary to get outputs on n *adaptively chosen* input indices of its choice before trying to predict the remaining output.[11] More details follow.

The definitions of the selective CI-pseudorandomness and adaptive notions of all the three notions are given in the full version.

The Selective Correlated-Input Predicting *experiment* $Exp^{sCI-pred}_{\mathcal{A},\mathcal{H},\{C\}}$. For a family of deterministic functions $\mathcal{H}$, an adversary $\mathcal{A}$, and a family of efficiently-computable correlated-input circuits $\{C\}$, we define the following game between a challenger and the adversary $\mathcal{A}$.

– **Setup Phase 1**. Challenger runs the Gen algorithm of $\mathcal{H}$ for a security parameter input 1^λ and gets $h_c : \mathcal{D}_h \longrightarrow \mathcal{R}_h$. Challenger gives $\mathcal{D}_h$ to $\mathcal{A}$.

[11] By incurring a security loss of a factor of n, this definition can actually be shown to be equivalent to a weaker definition where the adversary is required to predict the output specifically on input m_{n+1} fixed after it presents its queries but before it reveives the responses. However, working directly with this definition might lead to better concrete security guarantees in the real world.

– **Query Phase**. $\mathcal{A}$ chooses a positive integer n ($= poly(\lambda)$), and gives to the challenger $n+1$ distinct circuits $\{C_i\}_{i\in[n+1]} \subset \{C\}$.
– **Setup Phase 2**. Challenger gives $h_c(\cdot)$ to $\mathcal{A}$ and chooses r, a uniform random string of appropriate length.
– **Partially Adaptive Query-Response Phase**. $\mathcal{A}$ presents n queries, where an ith query is $k_i \in [n+1]$. Challenger responds to it via $h_c(C_{k_i}(r))$.
– **Predict Phase**. The adversary outputs $\hat{y} \in \mathcal{R}_h$.

Let $k_{n+1} \in [n+1]$ be such that $k_{n+1} \neq k_i \ \forall \ i \in [n]$. The output of the experiment is defined to be 1 if $\hat{y} = h_c(C_{k_{n+1}}(r))$ and 0 otherwise.
We define the advantage of an adversary $\mathcal{A}$ in the above game as:

$$\mathsf{Adv}^{sCI-pred}_{\mathcal{A},\mathcal{H},\{C\}}(\lambda) = \Pr[\mathsf{Exp}^{sCI-pred}_{\mathcal{A},\mathcal{H},\{C\}} = 1]$$

The probability is over the random bits used by the challenger and the adversary.

Definition 6. *A family of functions $\mathcal{H}$ is said to be selective correlated-input unpredictable with respect to a family of correlated-input circuits $\{C\}$, if for all $\mathcal{A} \in PPT$ there exists a negligible function $negl$, such that:*

$$Adv^{sCI-pred}_{\mathcal{A},\mathcal{H},\{C\}}(\lambda) \leq negl(\lambda)$$

4 Proposed Construction

In the sequel, we give the construction of our function and prove that it is correlated-input secure for a class of polynomials over $\mathbb{Z}_p$ (where p is a prime number) in the sense of each of the three selective security models defined above.

Our construction is given in Figure 1.

$\mathsf{Gen}(1^\lambda)$. Run GrpGen: $(\mathbb{G}, p, g) \leftarrow \mathsf{GrpGen}(1^\lambda)$, where p is a prime number. Gen uniformly samples a random element c from $\mathbb{Z}_p$ and a random generator g of group $\mathbb{G}$. It outputs g, c and a function $h : \mathbb{Z}_p \longrightarrow \mathbb{G}$ defined by,

$$h(m) := g^{\frac{1}{m+c}}$$

for any $m \in \mathbb{Z}_p$ (where $\frac{1}{m+c}$ is computed $\mod p$).

Fig. 1. Our Construction

Our proposed function is extremely simple and efficient to compute. The cost of computation is dominated by a single exponentiation operation. The construction can be seen as similar to a short signature scheme by Boneh and Boyen [10]. Our main novelty can be seen in the proofs of security. Indeed, interestingly, our

proofs show that the original signature scheme of Boneh and Boyen is secure even if an adversary is allowed to obtain messages signed by various *correlated secret signing keys*, where the correlations are from a set of polynomials over $\mathbb{Z}_p$. We refer the reader to the full version for a detailed description of this implication.

4.1 Analysis of the above Construction.

We prove that our construction is selectively secure against a class of correlations computable by polynomials over $\mathbb{Z}_p$. In what follows, we introduce some more notations that will be used in the rest of the paper and then discuss the three security games with correlated-input circuits computing polynomials over $\mathbb{Z}_p$.

Notations Let $deg[f(X)]$ denote the degree of a polynomial $f(X)$ over $\mathbb{Z}_p$. If the output distribution of a polynomial is uniform in $\mathbb{Z}_p$ (or, in other words, if the range of the polynomial is $\mathbb{Z}_p$ itself), then we refer to the polynomial as uniform-output polynomial. On the other hand, if the output distribution of a polynomial has high min-entropy in $\mathbb{Z}_p$, then we refer to the polynomial as high-min-entropy-output polynomial (any non-zero polynomial of degree polynomial in the security parameter has a range of size exponential in the security parameter). We shall only consider polynomials of degree is at least 1 and polynomial in λ. We only state our theorems in the following and give the proofs in the full version.

4.2 Selective Correlated-Input one-wayness

Theorem 1. *Suppose that (q, t', ϵ)-SDL assumption holds in $\mathbb{G}$. Let $\{C\}$ be a set of non-zero polynomials over $\mathbb{Z}_p$. Then, for $\mathcal{H}$ as in Figure 1, there exists no probabilistic t-time adversary $\mathcal{A}$ for which $\mathsf{Adv}^{sCI-inv}_{\mathcal{A},\mathcal{H},\{C\}}(\lambda)$ is at least ϵ provided that*

$$d \leq q \text{ and } t \leq t' - \Theta(nq\tau)$$

where $d = d(\lambda)$ upper bounds the sum of the degrees of the polynomials that $\mathcal{A}$ queries upon and τ is the maximum time for an exponentiation in $\mathbb{G}$ and $\mathbb{Z}_p$.

4.3 Selective Correlated-Input Unpredictability

Theorem 2. *Suppose that (q, t', ϵ')-DHI assumption holds in $\mathbb{G}$. Let $\{C\}$ be a set of uniform-output polynomials over $\mathbb{Z}_p$. Then, for $\mathcal{H}$ as in Figure 1, there exists no probabilistic t-time adversary $\mathcal{A}$ for which $\mathsf{Adv}^{sCI-pred}_{\mathcal{A},\mathcal{H},\{C\}}$ is at least ϵ provided that*

$$d \leq q+1,\ \epsilon \geq 2(n+1)\epsilon' \text{ and } t \leq t' - \Theta(nq\tau)$$

where $d = d(\lambda)$ upper bounds the sum of the degrees the polynomials that $\mathcal{A}$ queries upon and τ is the maximum time for an exponentiation in $\mathbb{G}$ and $\mathbb{Z}_p$.

The proof for the above theorem and for CI-pseudorandomness is given in the full version.

5 Relations between CI-Security and RKA-Security

We conclude by examining relations beween correlated-input secure hash functions and security under related-key attacks, whose formal treatement was initiated by [5]. The latter asks that security of a cryptographic primitive (e.g., a pseudorandom function) maintains security when used with *related secret keys*. In this section, we only consider our notion of pseudorandomness under correlated-inputs, so "CI-security" below refers by default to this notion.

5.1 Relations to RKA-Secure Weak PRFs

We start by showing an equivalence between CI-secure hash functions and some form of RKA-security for *weak* PRFs we introduce. The idea, following [20], is to "switch" the input and key for these primitives.

RKA-Secure Weak PRFs Recall that weak PRFs [23], as opposed to normal ones, handle only random inputs. In defining RKA-security for this primitive, a modeling choice we need to consider is whether, when the adversary queries for a value of the function under a related key, a *new* random input is chosen (or the previous random-input re-used). We give general definitions that capture the possibilities. (However, for our results we only use a notion where the same random input is re-used.) We also consider both "selective" and "adaptive" security; in the former, the adversary chooses the relations applied to the secret key before receiving any responses.

In defining RKA security for various primitives, we use a non-standard formalization based on our framework of circuits, which in this case sample keys. This is for ease of comparison to our notion of CI-security. For example, by $\{C\}$-RKA-PRF we refer to an RKA-PRF where the secret keys are sampled according to circuits in $\{C\}$ (executed on a common random input). Note that, in this framework, RKA-security corresponds to a special case of CI-security where the first circuit samples a random key K and and the remaining circuits operate only on K (and not the coins used to sample it) to produce a related key. (The latter is sufficient in the context of RKA-security.)

We also note that a PRF function family is specified by an efficient probabilistic parameter-generation algorithm Gen_{prf} which takes as input a security parameter 1^λ and outputs the description of a function including a description of its keyspace, domain and range. However, for simplicity of exposition, we only consider a single PRF function in most part of the following discussion as long as there is no ambiguity.

Definition 7 ($q_{input} - \{C\}$**-aRKA-wPRF.).** *Let $F : \mathcal{K} \times \mathcal{D} \longrightarrow \mathcal{R}$ be an efficiently computable function. Let $\{C\} \subseteq \mathsf{Fun}(\mathcal{K}, \mathcal{K})$ be a set of RKD circuits. F is said to be $q_{input} - \{C\}$-aRKA-wPRF, if, $\forall \mathcal{A} \in \mathsf{PPT}$:*
$\mathsf{Adv}^{aRKA-wPRF}_{\mathcal{A},F,\{C\},q_{input}}(\lambda) := |\Pr[k \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{O}^{weak}_{F(k,\cdot)}(\cdot,\cdot)} \longrightarrow 1] - \Pr[k \xleftarrow{\$} \mathcal{K}, G \xleftarrow{\$} \mathsf{FF}(\mathcal{K}, \mathcal{D}, \mathcal{R}) : \mathcal{A}^{\mathcal{O}^{weak}_{G(k,\cdot)}(\cdot,\cdot)} \longrightarrow 1]|$
is negligible in λ, where the related-key-wprf oracle $\mathcal{O}^{weak}_{f(k,\cdot)}(\cdot)$ takes as input

$(index_i, C_i) \in ([q_{input}], \{C\})$, *and outputs* $(x_{index_i}, f(C_i(k), x_{index_i}))$, *where* $x_{index_i} \xleftarrow{\$} \mathcal{D}$.

The definition for selective version appears in the full version.

The Equivalence In the following, for a class of circuits $\{C\}$ we show an equivalence between a one-input RKA-Secure wPRF for $\{C\}$ and a CI-secure hash function for $\{C\}$. (The equivalence holds respectively in the cases of selective and adaptive security notions.) First, we construct a family of functions $\{F\}$ from a family of functions $\mathcal{H}$ and show that if $\mathcal{H}$ is a $\{C\}$-aCI-pseudorandom function family (resp. $\{C\}$-sCI-pseudorandom function family) then $\{F\}$ is a $1-\{C\}$-aRKA-wPRF (resp. $1-\{C\}$-sRKA-wPRF).

Let $\mathcal{H}$ be a family of functions specified by Gen. A family of functions $\{F\}$ is defined by the following parameter-generation algorithm:
$\mathsf{Gen}_{prf}(1^\lambda)$. Gen_{prf} runs $\mathsf{Gen}(1^\lambda)$ that outputs the description of a parameter set $\mathcal{D}$, $c \in \mathcal{D}$, and a function $h_c : \mathcal{K} \longrightarrow \mathcal{R}$. Gen_{prf} outputs $\mathcal{K}$, $\mathcal{D}$ and $\mathcal{R}$ for keyspace, domain and range, respectively, of a function F defined by,

$$F(k, x) := h_x(k)$$

for any $k \in \mathcal{K}$ and $x \in \mathcal{D}$.

Fig. 2. Construction of $1-\{C\}$-(s/a)RKA-wPRF Family from $\{C\}$-(s/a)CI-pseudorandom Function Family

Theorem 3. *$\{C\}$-aCI-pseudorandom function family (resp. $\{C\}$-sCI-pseudorandom function family) implies $1-\{C\}$-aRKA-wPRF (resp. $1-\{C\}$-sRKA-wPRF).*

The proof is given in the full version.

In the full version, we also give a construction of $1-\{C\}$-aRKA-wPRF (resp. $1-\{C\}$-sRKA-wPRF) from $\{C\}$-aCI-pseudorandom function family (resp. $\{C\}$-sCI-pseudorandom function family).

New CI-Secure Hash Functions As an application of the above equivalence, we obtain new CI-secure hash functions. In particular, note that an adaptive one-input RKA-secure wPRF for a class of circuits $\{C\}$ is trivially implied by an RKA-Secure PRF for $\{C\}$. (Here, the latter is defined as expected, namely as in prior work except cast in our framework of circuits; see the full version for more details.) We can therefore use the recent constructions of RKA-Secure PRFs by Bellare and Cash [3] to obtain adaptive CI-secure hash functions. Namely, the latter are secure under the standard DDH assumption for class of circuits computing multiplication by a group element or under exponential-hardness of DDH for addition by a group element.

Though the resulting CI-secure hash functions are secure for much weaker classes of relations as compared to our main construction, they are remarkable in that they are both adaptively secure and do not need a public key. (They do

not even need any randomly-generated global parameters, as the constructions of [3] work in a fixed group.) The latter is because in the case that we start with an RKA-secure PRF (rather than wPRF), our construction of CI-secure hash can be modified by applying the PRF to any fixed value in the domain of the latter (still using the input as the key).

5.2 CI-secure Functions Imply Other RKA-secure Primitives

In this section, we discuss a more general technique for building RKA-secure cryptographic primitives from a CI-secure hash function. The basic idea is to hash the coins used to generate keys for the the former, using a CI-secure hash function.

Informally, let Ψ be a scheme for a cryptographic primitive. Let KeyGen be a PPT algorithm for Ψ, and let $l(\lambda)$ be the length of the random string used by it. Our transformation uses a $\{C\}$-pseudorandom function family $\mathcal{H}$ specified by Gen, and the transformation involves modifying $\mathsf{KeyGen}(1^\lambda; r)$ where $r \xleftarrow{\$} \{0,1\}^{l(\lambda)}$ to $\mathsf{KeyGen}(1^\lambda; h(r'))$ where $r' \xleftarrow{\$} \{0,1\}^{t(\lambda)}$ and $(h : \{0,1\}^{t(\lambda)} \longrightarrow \{0,1\}^{l(\lambda)}) \leftarrow \mathsf{Gen}(1^\lambda)$. The resulting scheme is then expected to be " $\{C\}$-RKA-secure" besides preserving the security properties of the underlying untransformed scheme.

More concretely, we exemplify the above technique for digital signatures. We give our formalization of RKA-security for signatures in the full version.

In what follows, we show that $\{C\}$-aCI-pseudorandom function family (resp. $\{C\}$-sCI-pseudorandom function family) implies $\{C\}$-aRKA-unforgeable scheme (resp. $\{C\}$-sRKA-unforgeable scheme). The transformation is given in Figure 3.

Let $\mathcal{H}$ be a function family specified by Gen. Let $\Sigma' = (\mathsf{KeyGen}', \mathsf{Sign}', \mathsf{Verify}')$ be a signature scheme and let $l(\lambda)$ be the length of the randomness used in the KeyGen'. The signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is defined by:

- $\mathsf{KeyGen}(1^\lambda)$: $(h : \{0,1\}^{t(\lambda)} \longrightarrow \{0,1\}^{l(\lambda)}) \leftarrow \mathsf{Gen}(1^\lambda)$; $sk \xleftarrow{\$} \{0,1\}^{t(\lambda)}$; $(sk', pk') \leftarrow \mathsf{KeyGen}'(1^\lambda; h(sk))$; output sk as the secret key, and $pk := (h, pk')$ as the public key.
- $\mathsf{Sign}(sk, m)$: Run $\mathsf{KeyGen}'(1^\lambda, h(sk))$ to obtain sk'. The signature on message m is set as $\sigma \leftarrow \mathsf{Sign}'(sk', m)$.
- $\mathsf{Verify}(pk, m, \sigma)$: Output valid if $\mathsf{Verify}'(pk', m, \sigma) = \mathsf{valid}$ and output $\mathsf{invalid}$ otherwise.

Fig. 3. Construction of RKA-secure Signature Scheme from CI-secure Pseudorandom Functions

Theorem 4. *$\{C\}$-aCI-pseudorandom function family (resp. $\{C\}$-sCI-pseudorandom function family) implies $\{C\}$-aRKA-unforgeable scheme (resp. $\{C\}$-sRKA-unforgeable scheme).*

The proof is given in the full version.

Discussion In the case that the starting CI-secure hash function has a public key, the above transformation results in a cryptographic primitive for which algorithms operating on the secret key also need to access an authentic public key. In some scenarios, e.g. smart cards, this may not always be practical. Moreover, our main construction of CI-secure hash is only selectively-secure, resulting in a selectively RKA-secure the cryptographic primitive. On the other hand, it is sometimes possible to use our techniques (in a "non-blackbox" way) to design an RKA-secure scheme without a public key and that is adaptively secure. In particular, we show this for RKA-secure symmetric-key encryption recently introduced in [1] in the full version. We also mention that using the CI-secure hash functions derived from the Bellare-Cash RKA-secure PRFs [3] avoid these issues, but for a much weaker class of relations.

Relation to Tampering Attacks We also note that RKA-security for a cryptographic primitive can also be used to to protect against tampering attacks [18], where, for instance, the secret key stored by a smart card is tampered with and its behavior is observed while it acts using the tampered secret, with an objective of gaining advantage against the security of the functionality of the smart card when using the original secret. However, as discussed in [1], security against tampering attacks is easier to achieve in general, through some kind of "sanity check" on the secret key (for instance, by including a signature on the secret key as a part of the public key, which is verified by any algorithm using the former); although, as discussed above, this approach may not always be practical. This does not work for RKA-security, since we actually want related secret keys to function like *independently generated* ones.

Acknolwedgements

We thank the anonymous reviewers of TCC 2011 for detailed comments and suggestions. The second author thanks David Cash for discussions about related-key security, and was supported in part by Brent Waters' grants NSF CNS-0915361 and CNS-0952692.

References

1. Applebaum, B., Harnik, D., Ishai, Y.: Semantic security under related-key attacks and applications. Cryptology ePrint Archive, Report 2010/544 (2010), http://eprint.iacr.org/
2. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: CRYPTO. pp. 535–552 (2007)
3. Bellare, M., Cash, D.: Pseudorandom functions and permutations provably secure against related-key attacks. In: CRYPTO. pp. 666–684 (2010)
4. Bellare, M., Fischlin, M., O'Neill, A., Ristenpart, T.: Deterministic encryption: Definitional equivalences and constructions without random oracles. In: CRYPTO. pp. 360–378 (2008)

5. Bellare, M., Kohno, T.: A theoretical treatment of related-key attacks: Rka-prps, rka-prfs, and applications. In: EUROCRYPT. pp. 491–506 (2003)
6. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security. pp. 62–73 (1993)
7. Boldyreva, A., Cash, D., Fischlin, M., Warinschi, B.: Foundations of non-malleable hash and one-way functions. In: ASIACRYPT. pp. 524–541 (2009)
8. Boldyreva, A., Fehr, S., O'Neill, A.: On notions of security for deterministic encryption, and efficient constructions without random oracles. In: CRYPTO. pp. 335–359 (2008)
9. Boneh., D., Boyen, X.: Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles. In: Advances in Cryptology – Eurocrypt. LNCS, vol. 3027, pp. 223–238. Springer (2004)
10. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT. Lecture Notes in Computer Science, vol. 3027, pp. 56–73. Springer (2004)
11. Boneh, D., Crescenzo, G.D., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: EUROCRYPT. pp. 506–522 (2004)
12. Canetti, R.: Towards realizing random oracles: Hash functions that hide all partial information. In: CRYPTO. pp. 455–469 (1997)
13. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)
14. Canetti, R., Micciancio, D., Reingold, O.: Perfectly one-way probabilistic hash functions (preliminary version). In: STOC. pp. 131–140 (1998)
15. Coppersmith, D., Franklin, M.K., Patarin, J., Reiter, M.K.: Low-exponent rsa with related messages. In: EUROCRYPT. pp. 1–9 (1996)
16. Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: FOCS. pp. 293–302 (2008)
17. Feistel, H.: Cryptography and computer privacy. Scientific American 228(5) (1973)
18. Gennaro, R., Lysyanskaya, A., Malkin, T., Micali, S., Rabin, T.: Algorithmic tamper-proof (atp) security: Theoretical foundations for security against hardware tampering. In: TCC. pp. 258–277 (2004)
19. Goldenberg, D., Liskov, M.: On related-secret pseudorandomness. In: TCC. pp. 255–272 (2010)
20. Hemenway, B., Lu, S., Ostrovsky, R.: Correlated product security from any one-way function and the new notion of decisional correlated product security. Cryptology ePrint Archive, Report 2010/100 (2010), http://eprint.iacr.org/
21. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: CRYPTO. pp. 145–161 (2003)
22. Kiltz, E., Pietrzak, K., Stam, M., Yung, M.: A new randomness extraction paradigm for hybrid encryption. In: EUROCRYPT. pp. 590–609 (2009)
23. Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of pseudo-random functions. J. Comput. Syst. Sci. 58(2), 336–375 (1999)
24. O'Neill, A.: Deterministic public-key encryption revisited. Cryptology ePrint Archive, Report 2010/533 (2010), http://eprint.iacr.org/
25. Rosen, A., Segev, G.: Chosen-ciphertext security via correlated products. In: TCC. pp. 419–436 (2009)
26. Shannon, C.E.: Communication theory of secrecy systems. Bell System Technical Journal 28(4), 656–715 (1949)
27. Wagner, D., Goldberg, I.: Proofs of security for the unix password hashing algorithm. In: ASIACRYPT. pp. 560–572 (2000)