Journal of the Nigerian Association of Mathematical Physics
Volume **12 (May, 2008), 495 - 504**

Universal hash proof systems: A construction for a provably-secure public-key encryption scheme

Tola John Odule,
Department of Mathematical Sciences
Olabisi Onabanjo University, Ago-Iwoye, Ogun State, Nigeria

Abstract

This paper presents a new and fairly practical public-key encryption scheme proven secure against adaptive chosen ciphertext attack. The scheme is based on Decision Composite Residuosity (DCR) assumption. The analysis of the scheme is in the standard cryptographic model, and does not rely on the Random Oracle model. We show how to construct an efficient public-key encryption schemes secure against adaptive chosen ciphertext attack in the standard model. Our construction only uses the universal hash proof system as a primitive: no other primitives are required, although even more efficient encryption schemes can be obtained by using hash functions with appropriate collision-resistance properties.

Keywords: Public-key Encryption, Decision Composite Residuosity, Adaptive-chosen Attack, Universal Hash Proof, Random Oracle.

1.0 Introduction

While there are weaker notions of security, such as that defined by Naor and Yung [1, 2], experience in the design and analysis of cryptographic protocols has shown that security against adaptive chosen ciphertext attack [3, 4] is both necessary and sufficient in many applications.

Until now, the only practical scheme that has been proposed that can be proven secure against adaptive chosen ciphertext attack under a reasonable intractability assumption is that of Cramer and Shoup [5, 6]. This scheme is based on the Decision Diffie-Hellman (DDH) assumption, and is not much less efficient than traditional ElGamal encryption.

Other practical schemes have been proposed and heuristically proved secure against adaptive chosen ciphertext. More precisely, these schemes are proven secure under reasonable intractability assumptions in the Random Oracle model [7]. While the Random Oracle model is a useful heuristic, it does not rule out all possible attacks: a scheme proven secure in this model might still be subject to an attack "in the real world," even though the stated intractability assumption is true, [8] and even if there are no particular weaknesses in the cryptographic hash function [9].

2.0 A general scheme for provably secure public-key encryption

In this section, we present a general technique for building secure public-key encryption schemes using appropriate hash proof systems for a hard subset membership problem. But first, we recall the definition of a public-key encryption scheme and the notion of security against adaptive chosen ciphertext attack.

2.1 Public-key encryption schemes

A public key encryption scheme provides three algorithms:

- a probabilistic, polynomial-time key generation algorithm that on input 1^{ℓ}, where $\ell \geq 0$ is a is

e-mail: tee_johnny@yahoo.com

security parameter, outputs a public-key/private-key pair (***PK, SK***)..

A public key ***PK*** specifies an finite message space M_{PK}. The message space should be easy to recognize; that is, there should be a deterministic, polynomial-time algorithm that takes as input 1^λ and ***PK***, along with a bit string ς, and determines if ς is a proper encoding of an element of M_{PK}.

- a probabilistic, polynomial-time encryption algorithm that on input 1^λ, ***PK,*** and *m*, where $\lambda \geq 0$, ***PK*** is a public key associated with security parameter λ, and $m \in M_{PK}$, outputs a bit string σ.
- *a* ***sdeterministic, polynomial-time*** decryption algorithm that on input 1^λ, ***SK***, and σ, where $\lambda \geq 0$, *SK* is a private key associated with security parameter λ, and σ is a bit string, outputs either a message $m \in M_{PK}$, where ***PK*** is the public-key corresponding to ***SK,*** or a special symbol reject.

Any public-key encryption scheme should satisfy a "correctness" or "soundness" property, which loosely speaking means that the decryption operation "undoes" the encryption operation. For our purposes, we can formulate this as follows. Let us call a key pair (***PK, SK***) bad if for some $m \in M_{PK}$, and for some encryption σ of *m* under ***P****K*, the decryption of σ under ***SK*** is not *m*. Let us call a public-key encryption scheme sound if the probability that the key generation algorithm on input 1^λ outputs a bad key pair is a negligible function in λ.

For all encryption schemes presented in this paper, it is trivial to verify this soundness property, and so we will not explicitly deal with this issue again. We only work with finite message spaces in this discourse.

2.2 Adaptive chosen ciphertext security

Consider a public-key encryption scheme, and consider the following game, played against an arbitrary probabilistic, polynomial-time adversary.

1. *Key-Generation Phase*. Let $\lambda \geq 0$ be the security parameter. We run the key-generation algorithm of the public-key encryption scheme on input 1^λ, and get a key pair (***PK, SK***). We equip an encryption oracle with the public key ***PK,*** and a decryption oracle with the secret key ***SK***. The public-key ***PK*** is presented to the adversary.
2. *Probing Phase I*. In this phase, the attacker gets to interact with the decryption oracle in an arbitrary, adaptive fashion. This phase goes on for a polynomial amount of time, specified by the adversary.

More precisely, in each round of this interaction, the adversary sends a *query* σ to the decryption oracle. A query is a bit string chosen by the adversary. The decryption oracle in turn runs the decryption algorithm on input of the secret key ***SK*** and the query σ, and responds to the query by returning the output to the adversary.

Note that a query is not required to represent an encryption (under ***PK***) of a message; a query can indeed be any string designed to probe the behaviour of the decryption oracle. The interaction is adaptive in the sense that the next query may depend on the history so far, in some way deemed advantageous by the adversary.

3. *Target-Selection Phase*. The adversary selects two messages m_0 and m_1 from the message space, and presents (m_0, m_1) to the encryption oracle. The encryption oracle selects a random $\beta \in \{0, 1\}$, and encrypts m_β under ***PK***. The resulting encryption σ^*, the *target ciphertext*, is presented to the adversary.
4. *Probing Phase II*. This phase is as Probing Phase I, the only difference being that the decryption oracle only responds to queries σ that are different from the target ciphertext σ^*.
5. *Guessing-Phase*. The adversary outputs a bit β'. The adversary is said to win the game if $\beta' = \beta$. We define the advantage (over random guessing) of the adversary as the absolute value of the difference of the probability that he wins and ½. A public key encryption scheme is said to be *secure*

against adaptive chosen ciphertext attack if for all polynomial time, probabilistic adversaries, the advantage in this guessing game is negligible as a function of the security parameter.

3.0 The generic encryption/decryption scheme

We now describe our generic method for constructing a secure public-key encryption scheme. Let ***M*** be a subset membership problem specifying a sequence $(I_\lambda)_{\lambda \geq 0}$ of instance distributions. We also need a strongly smooth hash proof system ***P*** for ***M***, as well as a strongly *universal*$_2$ extended hash proof system $\mathbf{P}'$ for ***M***. We discuss ***P*** and $\mathbf{P}'$ below in greater detail.

To simplify the notation, we will describe the scheme with respect to a fixed value $\lambda \geq 0$ of the security parameter, and a fixed instance description $\Lambda[X, L, W, R] \in [I_\lambda]$. Thus, it is to be understood that the key generation algorithm for the scheme generates this instance description, using the instance sampling algorithm provided by ***M***, and that this instance description is a part of the public key as well; alternatively, in an appropriately defined "multi-user setting," different users could work with the same instance description.

With Λ fixed as above, let $\mathbf{H} = (H, K, X, L, \prod, S, \alpha)$ be the projective hash family [10, 11] that **P** associates with Λ, and let $\mathbf{H}' = (H', K', X \times \prod, L \times \prod, \prod, S', \alpha')$ be the projective hash family that $\mathbf{P}'$ associates with Λ. We require that $\prod$ is an abelian group, for which we use additive notation, and that elements of $\prod$ can be efficiently added and subtracted.

We now describe the key generation, encryption, and decryption algorithms for the scheme, as they behave for a fixed instance description Λ, with corresponding projective hash families **H** and $\mathbf{H}'$, as above. The message space is $\prod$.

3.1 Key generation

Choose $k \in K$ and $k' \in K'$ at random, and compute $s = \alpha(k) \in S$ and $s' = \alpha'(k') \in S'$. Note that all of these operations can be efficiently performed using the algorithms provided by **P** and $\mathbf{P}'$. The public key is (s, s') and the private key is (k, k').

3.2 Encryption

To encrypt a message $m \in \prod$ under a public key as above, one does the following: Generate a random $x \in L$, together with a corresponding witness $w \in W$, using the subset sampling algorithm provided by **M**. Compute $\pi = H_k(x) \in \prod$, using the public evaluation algorithm for **P** on inputs s, x, and w. Compute $e = m + \pi \in \prod$. Compute $\pi' = H'_k(x, e) \in \prod'$, using the public evaluation algorithm for $\mathbf{P}'$ on inputs s', x, e and w. The ciphertext is (x, e, π').

3.3 Decryption

To decrypt a ciphertext $(x, e, \pi') \in X \times \prod \times \prod'$ under a secret key as above, one does the following:

Compute $\pi'' = H'_k(x, e) \in \prod'$, using the private evaluation algorithm for $\mathbf{P}'$ on inputs k', x, and e. Check whether $\pi' = \pi''$; if not, then output reject and halt. Compute $\pi = H_k(x) \in \prod$, using the private evaluation algorithm for **P** on inputs *k* and *x*. Compute $m = e - \pi \in \prod$, and output the message *m*. It is to be implicitly understood that when the decryption algorithm is presented with a ciphertext, this ciphertext is actually just a bit string, and that the decryption algorithm must parse this string to ensure that it properly encodes some $(x, e, \pi') \in X \times \prod \times \prod'$; if not, the decryption algorithm outputs reject and halts.

We remark that to implement this scheme, all we really need is a ½-universal HPS, since we can convert this into appropriate strongly smooth and strongly *universal*$_2$ HPS's using appropriate standard constructions. Indeed, the Leftover Hash construction in Lemma 4 (see appendix) gives us a strongly smooth HPS whose hash outputs are bit strings of a given length b and so we can take the group $\prod$ in the above construction to be the group of b -bit strings with "exclusive or" as the group operation.

***Theorem* 3.1**

The above scheme is secure against adaptive chosen ciphertext attack, assuming $\mathbf{M}$ *is a hard subset membership problem.*

Proof:

We show that the existence of an efficient adaptive chosen ciphertext attack with non-negligible advantage implies the existence of an efficient distinguishing algorithm that contradicts the hardness assumption for $\mathbf{M}$.

We define the following game between a *simulator* and an adversary that carries out an adaptive chosen ciphertext attack. The simulator takes as input 1^{ℓ}, for $\ell \geq 0$, along with Λ [*X, L, W, R*] $\in$ [I_ℓ] and $x^* \in X$. The simulator provides a "simulated environment" for the adversary as follows. In this description, **H** and $\mathbf{H}'$ are fixed as in the description above of the encryption scheme.

In the Key-Generation Phase, the simulator runs the key-generation as usual, using the given value of Λ. In both Probing Phases I and II, the simulator runs the decryption algorithm, as usual, using the secret key generated in the Key-Generation Phase.

In the Target-Selection Phase, the attacker presents messages m_0 and m_1 of his choice to the simulator. The simulator flips a random coin β, and computes the target ciphertext (x^*,e^*, π'^*), where x^* is the value input to the simulator, in the following way. It first computes $\pi^* = H_k(x^*)$. using the private evaluation algorithm for **P** on inputs k and x^*. It then computes $e^* = m_\beta + \pi^*$. Finally, it computes $\pi'^* = H'_k(x^*, e^*)$, using the private evaluation algorithm for $\mathbf{P}'$ on inputs k', x, and e.

In the Guessing Phase, the adversary outputs a bit β'. The simulator outputs 1 if $\beta = \beta'$, and 0 otherwise, after which, the simulator halts. For each value of the security parameter $\lambda \geq 0$, we consider the behaviour of this simulator/adversary pair in two different experiments. In the first experiment, the simulator is given (Λ, x^*) , where $\Lambda[X, L, W, R]$ is sampled from I_λ, and x^* is sampled at random from L; let T'_λ be the event that the simulator outputs a 1 in this experiment. In the second experiment, the simulator is given (Λ, x^*), where $\Lambda[X, L, W, R]$ is sampled from I_λ, and x^* is sampled at random from $X \setminus L$; let T_λ be the event that the simulator outputs a 1 in this experiment.

Let $AdvDist(\lambda) = \left|\Pr[T_\lambda] - \Pr[T'_\lambda]\right|$; that is, $AdvDist(\lambda)$ is the distinguishing advantage of our simulator. Let $AdvCCA(\lambda)$ be the adversary's advantage in an adaptive chosen ciphertext attack.

Our goal is to show that $AdvCCA(\lambda)$ is negligible, provided $AdvDist(\lambda)$ is negligible. To make the proof more concrete and the efficiency of the reduction more transparent, we introduce the following notation. We let $Q(\ell)$denote an upper bound on the number of decryption oracle queries made by the adversary; we assume that this upper bound holds regardless of the environment in which the adversary operates. Next, we suppose that **P** is $\varepsilon(\lambda)$-smooth with approximation error $\delta(\lambda)$, and that $\mathbf{P}'$ is $\varepsilon'(\lambda)$-$universal_2$ with approximation error $\delta'(\lambda)$. Also, we assume that the instance sampling algorithm for **M** has approximation error $t(\ell)$, and that the subset sampling algorithm for **M** has approximation error $t'(\ell)$.

Case *x€ *L***.

In this case, the simulation is perfect, except for the approximation errors introduced by the instance and subset sampling algorithms for **M**. Thus, we have

$$\left|\Pr[T_\lambda'] - 1/2\right| \geq AdvCCA(\lambda) - (\iota(\lambda) + \iota'(\lambda)). \qquad (3.1)$$

Case *x€/L**.

To analyze the behaviour of the simulator in this case, it is convenient to make a sequence of modifications to the simulator. We refer to the experiment run with the unmodified simulator as experiment 0, and to the experiments run with subsequent modifications as experiments 2.1, 2.2tc. Each of these experiments are best viewed as operating on the same underlying probability space; we define

the event $T_\lambda^{(i)}$, for $i \geq 0$, as the event that the simulator in experiment i outputs a 1. Note that unlike the original simulator, these modified simulators need not be efficiently implementable.

***Experiment* 3.1**

To define experiment 1, we modify the simulator as follows. We replace the projective hash family **H** that **P** associates with Λ with its idealization, which is an $\varepsilon(\ell)$-smooth projective hash family that is $\delta(\ell)$-close to **H**. We also replace the projective hash family $\mathbf{H}'$ that $\mathbf{P}'$ associates with Λ with its idealization, which is an $\varepsilon'(\lambda)$-*universal*$_2$ projective hash family that is $\delta'(\lambda)$-close to $\mathbf{H}'$. By definition, we have $\left|\Pr[T_\lambda^{(1)}] - \Pr[T_\lambda^{(0)}]\right| \leq \delta(\lambda) + \delta'(\lambda)$ (3.2)

To keep the notation simple, we refer to these idealized projective hash families as **H** and $\mathbf{H}'$ as well, and continue to use the notation established in the description of the encryption scheme for these two projective hash families.

***Experiment* 3.2**

In experiment 3.2 we modify the simulator yet again, so that in addition to rejecting a ciphertext $(x, e, \pi') \in X \times \prod \times \prod'$ if the decryption oracle also rejects the ciphertext if $x \notin L$. Let F_2 be the event in experiment 3.2 that some ciphertext $(x, e, \pi') \in X \times \prod \times \prod'$ with $x \notin L$ is rejected by the decryption oracle but $H'_k(x, e) = \pi'$. We claim that

$$\Pr[F_2] \leq Q(\lambda)\varepsilon'(\lambda) \qquad (3.3)$$

To prove (3.3), let us condition on a fixed value of $\Lambda[X, L, W, R]$ (which determines the projective hash families **H** and $\mathbf{H}'$), as well as fixed values of k, s', and the adversary's coins. These values completely determine the public key, and all the decryption queries of the adversary and the responses of the simulator in Probing Phase I, and also determine if the adversary enters the Target-Selection Phase, and if so, the corresponding values of m_0 and m_1. Consider any ciphertext $(x, e, \pi') \in X \times \prod \times \prod'$, with $x \notin L$, that is submitted as a decryption oracle query during Probing Phase I. In this conditional probability space, x, e and π' are fixed, whereas k' is still uniformly distributed over K', subject only to the constraint that $\alpha'(k') = s'$, where s' is fixed as above. Therefore, from the $\varepsilon'(\lambda)$-*universal*$_2$ property of $\mathbf{H}'$, the probability that $H'_k(x, e) = \pi'$ in this conditional probability space is at most $\varepsilon'(\lambda)$.

Now assume that in this conditional probability space, the adversary enters the Target-Selection Phase. Let us now further condition on fixed values of β and x^* (which determine π^* and e^*), as well as a fixed value of π'^*. These values completely determine all the decryption queries of the adversary and the responses of the simulator in Probing Phase II. Consider any ciphertext $(x, e, \pi') \in X \times \prod \times \prod'$, with $x \notin L$, that is submitted as a decryption oracle query during Probing Phase II.

1. Suppose that $(x,e) = (x^*, e^*)$. Since we must have $(x,e,\pi) \neq (x^*, e^*, \pi^*)$, it follows that $\pi \neq \pi^*$, and hence $H_k(x,e) = \pi$ with certainty.

Suppose that $(x,e) \neq (x^*, e^*)$. In this conditional probability space, *x*, *e*, and π are fixed, whereas k is still uniformly distributed over K, subject only to the constraint that $\alpha(k) = s$, and

2. $H_k(x^*, e^*) = \pi^*$, where s, x^*, e^*, and π^* are fixed as above. Therefore, from the $\varepsilon(\lambda)$-*universal*$_2$ property of $\mathbf{H}$, the probability that $H_k(x,e) = \pi$ in this conditional probability pace is at most $\varepsilon(\lambda)$.

The above arguments show that for any individual ciphertext $(x,e,\pi^*) \in X \times \prod \times \prod$, with $x \notin L$, hat is submitted to the decryption oracle, the probability that $H_k(x,e) = \pi$ is at most $\varepsilon(\lambda)$, from which the bound (3.3) immediately follows.

Note that experiments 1 and 2 proceed identically until event F_2 occurs. More precisely, $T_\lambda^{(2)} \wedge \neg F_2$ occurs if and only if $T_\lambda^{(1)} \wedge \neg F_2$ occurs, which implies that

$$\left|\Pr[T_\lambda^{(2)}] - \Pr[T_\lambda^{(1)}]\right| \leq \Pr[F_2] \tag{3.4}$$

***Experiment* 3.3**

In experiment 3, we modify the simulator yet again. This time, in the encryption oracle, instead of computing π^* as $H_k(x^*)$ as, the simulator sets $\pi^* = \pi'$, where $\pi' \in \prod$ is chosen at random. Now, let us condition on a fixed value of $\Lambda[X, L, W, R]$ (which determines the projective ash families **H** and $\mathbf{H}$), as well as fixed values of k, β, and the adversary's coins. In this conditional probability space, since the action of H_k on *L* is determined by s, and since the simulator rejects all ciphertexts (x,e,π) with $x \notin L$, it follows that the output of the simulator in experiment 3.2 is completely determined as a function of *x**, *s*, and $H_k(x^*)$, while the output in experiment 3.3 is determined as the same function of *x**, *s*, and π'. Moreover, by independence, the joint distribution of (k, *x**, π') does not change in passing from the original probability space to the conditional probability space. It now follows directly from the $\varepsilon(\ell)$-smooth property of **H** that

$$\left|\Pr[T_\lambda^{(3)}] - \Pr[T_\lambda^{(2)}]\right| \leq \varepsilon(\lambda) \tag{3.5}$$

it is evident from the definition of the simulator in experiment 3.3 that the adversary's output β in this experiment is independent of the hidden bit β; therefore,

$$\Pr[T_\lambda^{(3)}] = 1/2\,. \tag{3.6}$$

Putting it all together. Combining the relations (3.2) - (3.6), we see that

$$\left|\Pr[T_\lambda] - 1/2\right| \leq \delta(\lambda) + \varepsilon(\lambda) + \delta'(\lambda) + Q(\lambda)\varepsilon'(\lambda) \tag{3.7}$$

Combining the inequalities (3.1) and (3.7), we see that

$$AdvCCA(\lambda) \leq AdvDist(\lambda) + \delta(\lambda) + \varepsilon(\lambda) + \delta'(\lambda) + Q(\lambda)\varepsilon'(\lambda) + \iota(\lambda) + \iota'(\lambda). \tag{3.8}$$

from which the theorem immediately follows.

4.0 Prototype model implementation

We present our public-key encryption scheme secure against adaptive chosen ciphertext attack. This is derived from the general construction in the previous section.

The scheme is based on Paillier's Decision Composite Residuosity (DCR) assumption [12]. This is a practical public-key encryption scheme secure against adaptive chosen ciphertext attack under this assumption.

4.1 Dcr-based public-key encryption

4.1.1 Derivation

The DCR assumption. Let p, q, p', q' be distinct odd primes with $p = 2p' + 1$ and $q = 2q' + 1$, and where p' and q' are both λ bits in length. Let $N = pq$ and $N' = p'q'$. Consider the group $Z^*_{N^2}$ and the subgroup P of $Z^*_{N^2}$ consisting of all Nth powers of elements in $Z^*_{N^2}$.

Paillier's Decision Composite Residuosity (DCR) assumption is that given only N, it is hard to distinguish random elements of $Z^*_{N^2}$ from random elements of P.

To be completely formal, a sequence of bit lengths $\lambda(\ell)$, parameterized by a security parameter $\ell \geq 0$ is hereby specified, and to generate an instance of the problem for security parameter ℓ, the primes p' and q' should be distinct, random primes of length $\lambda = \lambda(\ell)$, such that $p = 2p' + 1$ and $q = 2q' + 1$ are also primes.

We refer to the primes p' and q' as Sophie Germain primes and p and q as strong primes. Although it has never been proven that there are infinitely many Sophie Germain primes, nevertheless, it is widely conjectured, and amply supported by empirical evidence, that the probability that a random λ-bit number is Sophie Germain prime is $\Omega(1/\lambda^2)$. It is taken that this conjecture holds, so that we can assume that problem instances can be efficiently generated.

Note that Paillier did not make the restriction to strong primes in originally formulating the *DCR* assumption. However, one needs the restriction to strong primes for technical reasons. Nevertheless, it is easy to see that the *DCR* assumption without this restriction implies the *DCR* assumption with this restriction, assuming that strong primes are sufficiently dense, as we are here.

4.2 Subset membership problem

We can decompose $Z^*_{N^2}$ as an internal direct product $Z^*_{N^2} = G_N \cdot G_{N'} \cdot G_2 \cdot T$, where each group G_r is a cyclic group of order r, and T is the subgroup of $Z^*_{N^2}$ generated by $(-1 \bmod N^2)$. This decomposition is unique, except for the choice of G_2 (there are two possible choices). For any $x \in Z^*_{N^2}$, we can express x uniquely as $x = x(G_N)x(G_{N'})x(G_2)x(T)$, where for each G_r $x(G_r) \in G_r$ and $x(T) \in T$. Note that the element $\xi = (1 + N \bmod N^2) \in Z^*_{N^2}$ has order N, i.e., it generates G_N, and that $\xi^* = (1 + aN \bmod N^2)$ for $0 \leq a < N$. Define the map, $\theta: Z^*_{N^2} \to \{\pm 1\}$; $(a \bmod N^2)\ \alpha\ (a \mid N)$; where $(\cdot \mid \cdot)$ is the Jacobi symbol. It is clear that θ is a group homomorphism.

Let X be the kernel of θ. It is easy to see that $X = G_N G_N T$, since $\left| Z^*_{N^2} / X \right| = 2$ and $T \subset X$. In particular, X is a cyclic group of order $2NN'$. Let L be the subgroup of Nth powers of X. Then evidently, $L = G_{N'} T$, and so is a cyclic group of order $2N'$. These groups X and L will define our *subset membership problem*.

Our instance description Λ will contain N, along with a random generator g for L. It is easy to generate such a g: choose a random $\mu \in Z^*_{N^2}$, and set $g = -\mu^{2N}$. With overwhelming probability, such a g will generate L; indeed, the output distribution of this sampling algorithm is $O(2^{-\lambda})$-close the uniform distribution over all generators.

Let us define the set of witnesses as $W = \{0, \ldots, \lfloor N/2 \} \rfloor$. We say $w \in W$ is a witness for $x \in X$ if $x = g^w$. To generate $x \in L$ at random together with a corresponding witness, we simply generate $w \in W$ at random, and compute $x = g^w$. The output distribution of this algorithm is not the uniform distribution over L, but one that is $O(2^{-\lambda})$-close to it.

This completes the description of our subset membership problem. The reason for using (X, L) instead of $(Z_{N^2}^*, P)$ is that $Z_{N^2}^*$ and P are not cyclic, which is inconvenient for a number of technical reasons.

Next, we argue that the DCR assumption implies that this subset membership problem is hard. Suppose we are given x sampled at random from $Z_{N^2}^*$ (respectively, P). If we choose $b \in \{0,1\}$ at random, then $x^2(-1)^b$ is uniformly distributed over X (respectively, L). This implies that distinguishing X from L is at least as hard as distinguishing $Z_{N^2}^*$ from P, and so under the DCR assumption, it is hard to distinguish X from L. It is easy to see that this implies that it is hard to distinguish $X \setminus L$ from L as well.

4.3 Hash proof systems

Now it remains to construct appropriate strongly smooth and strongly $universal_2$ HPS's for the construction in section 2. To do this, we first construct a diverse group system from which we can then derive the required HPS's.

Fix an instance description Λ, where Λ specifies an integer N, defining groups X and L as above, along with a generator g for L. Let $\mathcal{H} = \mathrm{Hom}(X, X)$ and consider the group system G $\mathbf{G} = (\mathcal{H}, X, L, X)$, where $\mathbf{G}$ is a diverse group system. Moreover, for $x \in X$, we have $\mathcal{T}(x) = \langle x(G_N) \rangle$; thus, for $x \in X \setminus L$, $\mathcal{T}(x)$ has order p, q, or N, according to whether $x(G_N)$ has order p, q, or N. For $k \in Z$, let $H_k \in \mathrm{Hom}(X, X)$ be the kth power map; that is, H_k sends $x \in X$ to $x^k \in X$. Let $K_* = \{0, \ldots, 2NN' - 1\}$, the correspondence $k \alpha H_k$ yields a bijection between K_* and $\mathrm{Hom}(X, X)$.

Consider the projective hash family $\mathbf{H}_* = (H, K_*, X, L, X, L, \alpha)$, where H and K_* are as in the previous paragraph, and α maps $k \in Z$ to $H_k(g) \in L$. Clearly, $\mathbf{H}_*$ is a projective hash family derived from $\mathbf{G}$, and so it is $2^{-\lambda}$-universal. From this, we can obtain a corresponding HPS $\mathbf{P}$; however, as we cannot readily sample elements from K_*, the projective hash family $\mathbf{H}$ that $\mathbf{P}$ associates with the instance description Λ is slightly different than $\mathbf{H}_*$; namely, we use the set $K = \{0, \ldots, \lfloor N^2/2 \rfloor\}$ in place of the set K_*, but otherwise, $\mathbf{H}$ and $\mathbf{H}_*$ are the same. It is readily seen that the uniform distribution on K_* is $O(2^{-\lambda})$-close to the uniform distribution on K, and so $\mathbf{H}$ and $\mathbf{H}_*$ are also $O(2^{-\lambda})$-close. It is also easy to verify that all of the algorithms that $\mathbf{P}$ should provide are available.

So we now have a $2^{-\lambda(\lambda)}$-universal HPS $\mathbf{P}$. We could easily convert $\mathbf{P}$ into a strongly smooth HPS by applying the Leftover Hash Lemma construction in Lemma 4 (Odule, 2008) to the underlying universal projective hash family $\mathbf{H}_*$. However, there is a much more direct and practical way to proceed, as we now describe.

For any s, $x \in X$, if k is chosen at random from K_*, subject to $\alpha(k) = s$, then $H_k(x)$ is uniformly distributed over a coset of $\tau(x)$ in X. As discussed above, $\tau(x) = \langle x(G_N) \rangle$, and so is a subgroup of G_N. Moreover, for random $x \in X \setminus L$, we have $\tau(x) \neq G_N$ with probability at most $2^{-\lambda+1}$.

Now define the map

$$\chi: Z_{N^2} \to Z_N, \ (a + bN \bmod N^{2)} \ \alpha \ (b \bmod N) \quad (0 \le a, b < N).$$

This map does not preserve any algebraic structure; however, the restriction of λ to any coset of G_N in X is a one-to-one map from that coset onto Z_N. To see this, let $x = (a + bN \bmod N^2) \in X$, , where $0 \le a, b < N$, and note that we must have $\gcd(a, N) = 1$ for $0 \le c$, N, we have $x\xi^c = (\boldsymbol{a} + (ac + b)N \bmod N)$, and so $\chi(x\xi^c) = (ac + b \bmod N)$. For a, b fixed as above, as c ranges over $\{0, \ldots N - 1\}$, we see that $(ac + b \bmod N)$ ranges over Z_N.

Let us define $\mathbf{H}_*^{\times} = (H^{\times}, K_*, X, L, Z_N, L, \alpha)$, where for $k \in Z$, $H_k^{\times} = \lambda \circ H_k$. That is, $\mathbf{H}_*^{\times}$ is the same as $\mathbf{H}_*$, except that in $\mathbf{H}_*^{\times}$, we pass the output of the hash function for $\mathbf{H}_*$ through χ. From the observations in the previous two paragraphs, it is clear that $\mathbf{H}_*^{\times}$ is a $2^{-\lambda+1}$-smooth projective hash family. From $\mathbf{H}_*^{\times}$ we get a corresponding approximation $\mathbf{H}^{\times}$ (using K in place of K_*), and from this we get corresponding $2^{-\lambda(\lambda)+1}$-smooth HPS $\mathbf{P}^{\times}$.

We can then apply standard construction to $\mathbf{H}_*$, obtaining a $2^{-\lambda}$-$universal_2$ projective hash family $\acute{\mathbf{H}}_*$ for $(X \times Z_N \times L \times Z_N)$. From $\acute{\mathbf{H}}_*$ we get a corresponding approximation $\acute{\mathbf{H}}$ (using K in place of K_*), and from this we get a corresponding $2^{-\lambda(\lambda)}$-$universal_2$ extended HPS $\acute{\mathbf{P}}$.

We could build our encryption scheme directly using $\acute{\mathbf{P}}$; however, we get more compact ciphertexts if we modify $\acute{\mathbf{H}}_*$ by passing its hash outputs through χ, just as we did in building $\mathbf{H}_*^{\times}$, obtaining the analogous projective hash family $\acute{\mathbf{H}}_*^{\times}$ for $(X \times Z_N \times L \times Z_N)$. From this, it is then clear that $\acute{\mathbf{H}}_*^{\times}$ is also $2^{-\lambda}$-$universal_2$. From $\acute{\mathbf{H}}_*^{\times}$ we get a corresponding approximation $\acute{\mathbf{H}}^{\times}$ (using K in place of K_*), and from this we get a corresponding $2^{-\lambda(\lambda)}$-$universal_2$ extended HPS $\acute{\mathbf{P}}^{\times}$.

4.4 Encryption scheme

We now present in detail the encryption scheme obtained from the HPS's $\mathbf{P}^{\times}$ and $\acute{\mathbf{P}}^{\times}$ above. We describe the scheme for a fixed value of N that is the product of two $(\lambda+1)$-bit strong primes. The message space for this scheme is Z_N.

Let X, L, θ and λ be as defined above. Also, let $W = \{0,\ldots,[N/2]\}$ and $K = \{0,\ldots, [N^2/2]\}$, as above. Let $R = \{0\text{K}, 2^{\lambda} - 1\}$, and let $\Gamma : Z_{N^2} \times Z_N \to R^n$ be an efficiently computable injective map for an appropriate $n \ge 1$. For sufficiently large λ, $n = 7$ suffices.

4.4.1 Key Generation

Choose $\mu \in Z_{N^2}^*$ at random and set $\mathsf{g} = -\mu^{2\mathsf{N}} \in L$. Choose $k, \acute{k}, \acute{k}_1, \text{K}, \acute{k}_n \in K$ at random, and compute $s = g^k \in L$, $\acute{s} = g^{\acute{k}} \in L$, $\acute{s}_i = g^{\acute{k}_i} \in L \quad (i = 1, \text{K}, n)$. The public key is $(g; s : \acute{s} : \acute{s}_1, \text{K}, \acute{s}_n)$. The private key is $(k; \acute{k}; \acute{k}_1, \text{K}, \acute{k}_n)$

4.4.2 Encryption

To encrypt a message $m \in Z_N$ under a public key as above, one does the following. Choose $w \in W$ at random, and compute $x = g^w \in L$, $y = s^w \in L$, $\pi = \lambda(y) \in Z_N$, $e = m + \pi \in Z_N$. Compute $y = s^w \prod_{s=1}^{n} s_1^{nw} \in L$, $\pi = \lambda(y) \in Z_N$; where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$. The ciphertext is (x, e, π).

4.4.3 Decryption

To decrypt a ciphertext $(x, e, \pi) \in X \times Z_N \times Z_N$ under a secret key as above, one does the following:

Compute $y = x^{k + \sum_{i=1}^{n} \gamma_i k_1} \in X$, $\pi' = \lambda(y) \in Z_N$; where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$. Check whether $\pi = \pi'$; if not, then output reject and halt.

Compute $y = x^k \in X$, $\pi = \lambda(y) \in Z_N$, $m = e - \pi \in Z_N$ and output *m*.

Note that in the decryption algorithm, we are assuming that $x \in X$, which implicitly means that the decryption algorithm should check that $x \in Z_{N^2}^*$ and that $\theta(x) = 1$, and reject the ciphertext if this does not hold.

This is precisely the scheme that our general construction in section 2 yields. Thus, the scheme is secure against adaptive chosen ciphertext attack, provided the DCR assumption holds.

5.0 Conclusion

The DCR-based scheme is very practical. It uses an *n*-bit RSA modulus *N* (with, say, *n* = 1024). The public and private keys, as well as the ciphertexts, require storage for $O(n)$ bits. Encryption and decryption require $O(n)$ multiplications modulo N^2. Note that in this scheme, the factorization of *N* is not a part of the private key. This would allow, for example, many parties to work with the same modulus *N*, which may be convenient in some situations. Alternatively, if we include the factorization of *N* in the private key, some optimizations in the decryption algorithm are possible, such as Chinese Remaindering techniques.

APPENDIX

Let $\mathbf{H}^* = (H^*, K \times K, X, L, \prod, S \times K, \alpha^*)$, where H^* and α^* are defined as follows. For $k \in K$ $k \in K$, $k \in K$, and $x \in X$, we define $H^*_{k,k}(x) = H_k(H_k(x))$, and we define $\alpha^*(k, k) = (\alpha(k), k)$.

***Lemma* 4**

Let **H**, **F H***, and a be as in the above construction. Suppose that **H** is an ε-universal projective hash family. For any integer $b \geq 0$ such that $a + 2b \leq \log_2(1/\varepsilon)$, , **H*** is a $2^{-(b+1)}$-smooth projective hash family.

Proof:

It is clear that **H*** satisfies the basic requirements of a projective hash family. Consider the random variables $U(\mathbf{H}^*)$ and $V(\mathbf{H}^*)$. That is, consider the probability space where $x \in K$, $k \in K$, $x \in X \backslash L$, and $\pi' \in \prod$ are chosen at random, and set $U(\mathbf{H}^*) = (x, s, k, \pi')$ and $V(\mathbf{H}^*) = (x, s, k, \pi)$, where $s = \alpha(k)$ and $\pi = (H_k(H_k(x))$.

Consider any conditional probability space where particular values of $x \in X \backslash L$ and $s \in S$ are fixed, and let $U(\mathbf{H}^* | x, s)$ and $V(\mathbf{H}^* | x, s)$ be the random variables in this conditional probability space corresponding to $U(\mathbf{H}^*)$ and $V(\mathbf{H}^*)$. In such a conditional probability space, by the definition of $\in$-universal projective hashing, the distribution of $H_k(x)$ has min-entropy at least $\log_2(1/\varepsilon)$, and k is uniformly and independently

distributed over $\mathcal{K}$. The Leftover Hash Lemma then directly implies that $U(\mathbf{H}^*|x,s)$ and $V(\mathbf{H}^*|x,s)$ are $2^{-(b+1)}$-close. Since this bound holds uniformly for all x, s, it follows that $U(\mathbf{H}^*)$ and $V(\mathbf{H}^*)$ are also $2^{-(b+1)}$-close.

□

References

[1] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In Proc. STOC '90, ACM Press, 1990.

[2] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In Proc. STOC '89, ACM Press, 1989.

[3] Odule, T.J. "Incremental Cryptography and Security of Public Hash Functions." Journal of Nigerian Association of Mathematical Physics, vol. 11 pp.467-474; 2007.

[4] C. Racko_ and D. Simon. Non-interactive zero knowledge proof of knowledge and chosen-ciphertext attacks. In Proc. CRYPTO '91, Springer Verlag LNCS, 1991.

[5] R. Cramer and V. Shoup. A practical public key cryptosystem secure against adaptivechosen cipher text attacks. In Proc. CRYPTO '98, Springer Verlag LNCS, 1998.

[6] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. SIAM Journal on Computing, 30:391{437, 2000. Extended abstract in Proc. STOC '91, ACM Press, 1991.

[7] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing e_-cient protocols. In Proc. ACM Computer and Communication Security '93, ACM Press,1993.

[8] M. Luby. Pseudorandomness and Cryptographic Applications. Princeton University Press, 1996.

[9] R. Canetti, O. Goldreich, and S. Halevi. The random oracle model, revisited. In Proc.STOC '98, ACM Press, 1998.

[10] J. Carter and M. Wegman. Universal classes of hash functions. Journal of Computer andSystem Sciences, 18:143{154, 1979.

[11] M. Wegman and J. Carter. New hash functions and their use in authentication and set equality. Journal of Computer and System Sciences, 22:265{279, 1981.

[12] P. Paillier. Public-key cryptosystems based on composite degree residue classes. In Proc. EUROCRYPT '99, Springer Verlag LNCS, 1999.