# Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption

Ronald Cramer* Victor Shoup†

December 12, 2001

**Abstract**

We present several new and fairly practical public-key encryption schemes and prove them secure against adaptive chosen ciphertext attack. One scheme is based on Paillier's Decision Composite Residuosity (DCR) assumption, while another is based in the classical Quadratic Residuosity (QR) assumption. The analysis is in the standard cryptographic model, i.e., the security of our schemes does not rely on the Random Oracle model.

We also introduce the notion of a *universal hash proof system*. Essentially, this is a special kind of non-interactive zero-knowledge proof system for a language. We do not show that universal hash proof systems exist for all NP languages, but we do show how to construct *very efficient* universal hash proof systems for a general class of group-theoretic language membership problems.

Given an efficient universal hash proof system for a language with certain natural cryptographic indistinguishability properties, 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.

We show how to construct efficient universal hash proof systems for languages related to the DCR and QR assumptions. From these we get corresponding public-key encryption schemes that are secure under these assumptions. We also show that the Cramer-Shoup encryption scheme (which up until now was the only practical encryption scheme that could be proved secure against adaptive chosen ciphertext attack under a reasonable assumption, namely, the Decision Diffie-Hellman assumption) is also a special case of our general theory.

First version, October 12, 2001.

Second version, December 12, 2001: numerous minor notational changes and expositional improvements have been made, and additional and different variations on the basic DCR-based and QR-based schemes have been added.

*BRICS & Dept. of Computer Science, Aarhus University. Email: cramer@brics.dk

†IBM Zurich Research Laboratory. Email: sho@zurich.ibm.com

# 1 Introduction

It is generally considered that the “right” notion of security for security for a general-purpose public-key encryption scheme is that of *security against adaptive chosen ciphertext attack*.

This notion was introduced by Rackoff and Simon [RS]. While there are weaker notions of security, such as that defined by Naor and Yung [NY2], experience in the design and analysis of cryptographic protocols has shown that security against adaptive chosen ciphertext attack is both necessary and sufficient in many applications. Dolev, Dwork, and Naor [DDN] introduced the notion of *non-malleable encryption*, which turns out to be equivalent to the notion of security against adaptive chosen ciphertext attack (at least, when one considers the strongest possible type of adversary).

Although Rackoff and Simon defined the notion of security against adaptive chosen ciphertext attack, they did not actually present a scheme that satisfied this property. Indeed, although they present an encryption scheme, it requires the involvement of a *trusted third party* that plays a special role. Dolev, Dwork, and Naor present a scheme that can be proven secure against adaptive chosen ciphertext attack under a reasonable intractability assumption. However, although their scheme is polynomial time, it is horrendously impractical, and so although their scheme is a valuable proof of concept, it appears that it has no practical significance.

Up 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 [CS]. 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* [BR]. The Random Oracle model is an idealized model of computation in which a cryptographic hash function is modeled as a black box, access to which is allowed only through explicit oracle queries. 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, and even if there are no particular weaknesses in the cryptographic hash function (see [CGH]).

## 1.1 Our contributions

We present several new and fairly practical public-key encryption schemes and prove them secure against adaptive chosen ciphertext attack. One scheme is based on Paillier's Decision Composite Residuosity (DCR) assumption [P], while another is based in the classical Quadratic Residuosity (QR) assumption. The analysis is in the standard cryptographic model, i.e., the security of our schemes does not rely on the Random Oracle model.

We also introduce the notion of a *universal hash proof system*. Essentially, this is a special kind of non-interactive zero-knowledge proof system for a language. We do not show that universal hash proof systems exist for all NP languages, but we do show how to construct *very efficient* universal hash proof systems for a general class of group-theoretic language membership problems.

Given an efficient universal hash proof system for a language with certain natural cryptographic indistinguishability properties, 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.

We show how to construct efficient universal hash proof systems for languages related to the DCR and QR assumptions. From these we get corresponding public-key encryption schemes that are secure under these assumptions.

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 $N^2$.

The QR-based scheme is somewhat less practical. It uses an $n$-bit RSA modulus $N$ as above, as well as an auxiliary parameter $t$ (with, say, $t = 128$). The public and private keys require $O(nt)$ bits of storage, although ciphertexts require just $O(n + t)$ bits of storage. Encryption and decryption require $O(nt)$ multiplications modulo $N$.

We also show that the original Cramer-Shoup scheme follows from of our general construction, when applied to a universal hash proof system related to the DDH assumption.

### 1.1.1 Organization of the paper

The sections of this paper are organized as follows:

§2 recalls some basic terminology;

§3 recalls the classical notion of "universal hashing," and introduces a generalization which we call "universal projective hashing."

§4 formalizes the notion of a "subset membership problem";

§5 introduces the notion of a "universal hash proof system," which is based on "universal projective hashing," and "subset membership problems";

§6 presents a general framework for building a secure public-key encryption scheme using a "universal hash proof system" for a "hard subset membership problem."

§7 shows how to build practical "universal hash proof systems" for a general class of group-theoretic "subset membership problems."

§8 presents several new and fairly practical encryption schemes based on the preceding general constructions, including one based on the DCR assumption, and one based on the QR assumption, and also shows that the original Cramer-Shoup encryption scheme follows from these general constructions as well.

## 2 Some preliminaries

We recall some basic terminology and notation.

A function $f(\ell)$ mapping non-negative integers to non-negative reals if called *negligible (in $\ell$)* if for all $c \geq 1$, there exists $\ell_0 > 0$ such that $f(\ell) \leq 1/\ell^c$ for all $\ell \geq \ell_0$.

Let $X$ and $Y$ be random variables taking values in a finite set $S$. The *statistical distance* between $X$ and $Y$ is defined to be

$$\mathrm{Dist}(X, Y) = \frac{1}{2} \cdot \sum_{s \in S} |\Pr[X = s] - \Pr[Y = s]| \,.$$

Equivalently,

$$\mathrm{Dist}(X, Y) = \max_{S' \subset S} \left|\Pr[X \in S'] - \Pr[Y \in S']\right| \,.$$

We shall say that $X$ and $Y$ are $\epsilon$-close if $\mathrm{Dist}(X, Y) \leq \epsilon$.

Let $\mathbf{X} = (X_\ell)_{\ell \geq 0}$ and $\mathbf{Y} = (Y_\ell)_{\ell \geq 0}$ be sequences of random variables, where for each $\ell \geq 0$, $X_\ell$ and $Y_\ell$ take values in a finite set $S_\ell$. Then we say that $\mathbf{X}$ and $\mathbf{Y}$ are *statistically indistinguishable* if $\mathrm{Dist}(X_\ell, Y_\ell)$ is a negligible function in $\ell$. For computational purposes, we will generally work in a setting where the sets $S_\ell$ can be encoded as bit strings whose length is polynomial in $\ell$. For any probabilistic algorithm $A$ that outputs 0 or 1, we define the *distinguishing advantage for $A$ (with respect to $\mathbf{X}$ and $\mathbf{Y}$)* as the function

$$\mathrm{Dist}_A^{\mathbf{X},\mathbf{Y}}(\ell) = \left| \Pr[A(1^\ell, X_\ell) = 1] - \Pr[A(1^\ell, Y_\ell) = 1] \right|.$$

Here, the notation $1^\ell$ denotes the unary encoding of $\ell$ as a sequence of $\ell$ copies of 1, and the probability is with respect to the random coin tosses of the algorithm $A$ and the distributions of $X_\ell$ and $Y_\ell$. We say that $\mathbf{X}$ and $\mathbf{Y}$ are *computationally indistinguishable* if for all probabilistic, polynomial-time $A$, the function $\mathrm{Dist}_A^{\mathbf{X},\mathbf{Y}}(\ell)$ is negligible in $\ell$.

For a positive integer $Z$, $\mathbb{Z}_N$ denotes the ring of integers modulo $N$, and $\mathbb{Z}_N^*$ denotes the corresponding multiplicative group of units. For $a \in \mathbb{Z}$, $(a \bmod N) \in \mathbb{Z}_N$ denotes the residue class of $a$ modulo $N$.

For an element $g$ of a group $G$, $\langle g \rangle$ denotes the subgroup of $G$ generated by $g$. Likewise, for a subset $U$ of $G$, $\langle U \rangle$ denotes the subgroup of $G$ generated by $U$.

# 3 Universal projective hashing

## 3.1 Universal hashing

Before defining universal projective hash functions, we recall some definitions relating to the classical notion of "universal hashing" [CW, WC].

Let $X$ and $\Pi$ be finite, non-empty sets. Let $H = (H_k)_{k \in K}$ be a collection of functions indexed by $K$, so that for every $k \in K$, $H_k$ is a function from $X$ into $\Pi$. Note that we may have $H_k = H_{k'}$ for $k \neq k'$. We call $\mathbf{F} = (H, K, X, \Pi)$ a *hash family*, and each $H_k$ a *hash function.*

**Definition 1** *Let $\mathbf{F} = (H, K, X, \Pi)$ be a hash family, and consider the probability space defined by choosing $k \in K$ at random.*

*We call $\mathbf{F}$* pair-wise independent *if for all $x, x^* \in X$ with $x \neq x^*$, it holds that $H_k(x)$ and $H_k(x^*)$ are uniformly and independently distributed over $\Pi$.*

Note that there are many well-known, and very simple constructions of pair-wise independent hash families.

## 3.2 Definition of universal projective hashing

We now introduce the concept of universal projective hashing. Let $\mathbf{F} = (H, K, X, \Pi)$ be a hash family. Let $L$ be a non-empty, proper subset of $X$. Let $S$ be a finite, non-empty set, and let $\alpha : K \to S$ be a function. Set $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$.

**Definition 2** $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$*, defined as above, is called a* projective hash family (for $(X, L)$) *if for all $k \in K$, the action of $H_k$ on $L$ is determined by $\alpha(k)$.*

In other words, for all $k \in K$, the value $\alpha(k)$ "encodes" the action of $H_k$ on $L$ (and possibly more than that), so that given $\alpha(k)$ and $x \in L$, the value $H_k(x)$ is uniquely determined.

**Definition 3** *Let* $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ *be a projective hash family, and let* $\epsilon \geq 0$ *be a real number. Consider the probability space defined by choosing* $k \in K$ *at random.*

*We say that* $\mathbf{H}$ *is* $\epsilon$-universal *if for all* $s \in S$, $x \in X \setminus L$, *and* $\pi \in \Pi$, *it holds that*

$$\Pr[H_k(x) = \pi \ \wedge \ \alpha(k) = s] \leq \epsilon \Pr[\alpha(k) = s].$$

*We say that* $\mathbf{H}$ *is* $\epsilon$-universal$_2$ *if for all* $s \in S$, $x, x^* \in X$, *and* $\pi, \pi^* \in \Pi$ *with* $x \notin L \cup \{x^*\}$, *it holds that*

$$\Pr[H_k(x) = \pi \ \wedge \ H_k(x^*) = \pi^* \ \wedge \ \alpha(k) = s] \leq \epsilon \Pr[H_k(x^*) = \pi^* \ \wedge \ \alpha(k) = s]$$

We will sometimes refer to the value of $\epsilon$ in the above definition as the *error rate* of $\mathbf{H}$.

Note that if $\mathbf{H}$ is $\epsilon$-universal$_2$, then it is also $\epsilon$-universal (note that $|X| \geq 2$).

We can reformulate the above definition as follows. Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and consider the probability space defined by choosing $k \in K$ at random. $\mathbf{H}$ is $\epsilon$-universal means that conditioned on a fixed value of $\alpha(k)$, even though the value of $H_k$ is completely determined on $L$, for any $x \in X \setminus L$, the value of $H_k(x)$ can be guessed with probability at most $\epsilon$. $\mathbf{H}$ is $\epsilon$-universal$_2$ means that in addition, for any $x^* \in X \setminus L$, conditioned on fixed values of $\alpha(k)$ and $H_k(x^*)$, for any $x \in X \setminus L$ with $x \neq x^*$, the value of $H_k(x)$ can be guessed with probability at most $\epsilon$.

### 3.2.1 Motivation

We now discuss the motivation for Definition 3. Let $\mathbf{H}$ be a projective hash family, and consider the following game played by an adversary.

At the beginning of the game, $k \in K$ is chosen at random, and the adversary is given $s = \alpha(k)$. Initially, the adversary has no other information about $k$, but during the course of the game, he is allowed to make a sequence of *oracle queries* to learn more about $k$.

There are two types of oracle queries. One type of oracle query is a *test query*: the adversary submits $x \in X$ and $\pi \in \Pi$ to the oracle, and the oracle tells the adversary whether or not $H_k(x) = \pi$. The other type of oracle query is an *evaluation query*: the adversary submits $x^* \in X$ to the oracle, and the oracle tells the adversary the value $\pi^* = H_k(x^*)$.

During the course of the game, the adversary is allowed to make an arbitrary number of *test queries*, but only one *evaluation query*. Moreover, after the *evaluation query*, he is not allowed to submit $(x^*, \pi^*)$ to the oracle in any subsequent *test queries*.

We say the adversary *wins the game* if he submits a *test query* $(x, \pi)$ with $x \in X \setminus L$ and $H_k(x) = \pi$.

That completes the description of the game. Note that in this game, the adversary's strategy is quite arbitrary, and need not be efficiently computable. Moreover, the strategy may be adaptive, in the sense that an oracle query made by the adversary may depend in an arbitrary way on all information available to the adversary at that time.

It is easy to see from the definition that if $\mathbf{H}$ is $\epsilon$-universal$_2$, then regardless of the adversary's strategy, he wins the game with probability at most $Q \cdot \epsilon$, where $Q$ is a bound on the number of *test queries* made by the adversary. Note that while this property is a consequence of the definition of $\epsilon$-universal$_2$, it is not necessarily equivalent to the definition of $\epsilon$-universal$_2$. In fact, this property suffices to prove the main results of this paper, and indeed, all we need is this property in the case where $x^*$ is chosen at random from $X \setminus L$, and where the adversary is computationally bounded.

### 3.2.2 Trivial constructions

Families satisfying Definition 3 are trivial to construct, at least from a combinatorial point of view. For instance, let $\mathbf{F} = (H, K, X, \Pi)$ be a pair-wise independent hash family, let $L$ be a non-empty, finite subset of $X$, and let $\pi_0 \in \Pi$. Then let $\mathbf{H} = (H', K, X, L, \Pi, S, \alpha)$, where for all $k \in K$ and $x \in X$, we define $H'_k(x) = \pi_0$ if $x \in L$, and $H'_k(x) = H_k(x)$, otherwise. We also define $S = \{\pi_0\}$ and $\alpha(k) = \pi_0$ for all $k \in K$. It is clear that $\mathbf{H}$ is a $1/|\Pi|$-universal$_2$ projective hash family. However, in our applications later on, we want these hash functions to be efficiently computable on all of $X$, even if $L$ is *hard to distinguish* from $X \setminus L$. Therefore, this trivial "solution" is not useful in our context.

## 3.3 Smooth projective hashing

We will need a variation of universal projective hashing, which we call *smooth projective hashing.*

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family. We define two random variables, $U(\mathbf{H})$ and $V(\mathbf{H})$, as follows. Consider the probability space defined by choosing $k \in K$ at random, $x \in X \setminus L$ at random, and $\pi' \in \Pi$ at random. We set $U(\mathbf{H}) = (x, s, \pi')$ and $V(\mathbf{H}) = (x, s, \pi)$, where $s = \alpha(k)$ and $\pi = H_k(x)$.

**Definition 4** *Let $\epsilon \geq 0$ be a real number. A projective hash family $\mathbf{H}$ is* $\epsilon$-smooth *if $U(\mathbf{H})$ and $V(\mathbf{H})$ are $\epsilon$-close.*

## 3.4 Approximations to projective hash families

Our definition of universal and universal$_2$ projective hash families are quite strong: so strong, in fact, that in many instances it is impossible to efficiently implement them. However, in all our applications, it is sufficient to efficiently implement a projective hash family that effectively *approximates* a universal or universal$_2$ projective hash family. To this end, we define an appropriate notion of *distance* between projective hash families.

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family. Consider the distribution defined by sampling $k \in K$ at random, and define the random variable $View(\mathbf{H}) = (H_k, \alpha(k))$. Note that $View(\mathbf{H})$ comprises the value of $H_k$ at *all* points $x \in X$.

**Definition 5** *Let $\delta \geq 0$ be a real number. Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ and $\mathbf{H}^* = (H^*, K^*, X, L, \Pi, S, \alpha^*)$ be projective hash families. We say that $\mathbf{H}$ and $\mathbf{H}^*$ are $\delta$-close if $View(\mathbf{H})$ and $View(\mathbf{H}^*)$ are $\delta$-close.*

Note that if $\mathbf{H}$ and $\mathbf{H}^*$ are $\delta$-close for some "small" value of $\delta$, and if $\mathbf{H}^*$ is $\epsilon$-universal or $\epsilon$-universal$_2$ for some "small" value of $\epsilon$, this *does not* imply that $\mathbf{H}$ is $\epsilon'$-universal or $\epsilon'$-universal$_2$ for any particularly small value of $\epsilon'$. However, if $\mathbf{H}$ and $\mathbf{H}^*$ are $\delta$-close and $\mathbf{H}^*$ is $\epsilon$-smooth, then it is clear that $\mathbf{H}$ is $(\epsilon + \delta)$-smooth.

## 3.5 Some elementary reductions

We show some elementary reductions among the various notions introduced. Most of the reductions given here are primarily theoretically motivated. Later on, in a specialized context, we present reductions that are considerably more efficient.

### 3.5.1 Reducing the error rate

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be an $\epsilon$-universal (respectively, -universal$_2$) projective hash family. The construction below reduces the error rate from $\epsilon$ to $\epsilon^t$, by simple $t$-fold "parallelization."

Let $t$ be a positive integer, and let $\bar{\mathbf{H}} = (\bar{H}, K^t, X, L, \Pi^t, S^t, \bar{\alpha})$, where $\bar{H}$ and $\bar{\alpha}$ are defined as follows.

For $\vec{k} = (k_1, \ldots, k_t) \in K^t$ and $x \in X$, we define $\bar{H}_{\vec{k}}(x) = (H_{k_1}(x), \ldots, H_{k_t}(x))$, and we define $\bar{\alpha}(\vec{k}) = (\alpha(k_1), \ldots, \alpha(k_t))$.

The proof of the following lemma is straightforward, and is left to the reader.

**Lemma 1** *Let $\mathbf{H}$ and $\bar{\mathbf{H}}$ be as in the above construction. If $\mathbf{H}$ is an $\epsilon$-universal (respectively, -universal$_2$) projective hash family, then $\bar{\mathbf{H}}$ is an $\epsilon^t$-universal (respectively, -universal$_2$) projective hash family.*

### 3.5.2 From universal projective to universal$_2$ projective

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be an $\epsilon$-universal projective hash family. The next construction turns $\mathbf{H}$ into an $\epsilon$-universal$_2$ projective hash family $\mathbf{H}^\dagger$ for $(X, L)$.

Let us assume that we have injective functions $\Gamma : X \to \{0,1\}^n$ and $\Gamma' : \Pi \to \{0,1\}^{n'}$ for some appropriately large positive integers $n$ and $n'$. Let $\mathbf{H}^\dagger = (H^\dagger, K^{2n}, X, L, \{0,1\}^{n'}, S^{2n}, \alpha^\dagger)$, where $H^\dagger$ and $\alpha^\dagger$ are defined as follows.

For $\vec{k} = (k_{1,0}, k_{1,1}, \ldots, k_{n,0}, k_{n,1}) \in K^{2n}$, and $x \in X$ with $\Gamma(x) = (\gamma_1, \ldots, \gamma_n) \in \{0,1\}^n$, we define

$$H^\dagger_{\vec{k}}(x) = \bigoplus_{i=1}^{n} \Gamma'(H_{k_{i,\gamma_i}}(x))$$

and

$$\alpha^\dagger(\vec{k}) = (\alpha(k_{1,0}), \alpha(k_{1,1}), \ldots, \alpha(k_{n,0}), \alpha(k_{n,1})).$$

Here, "$\bigoplus$" denotes the bit-wise "exclusive or" operation on $n'$-bit strings.

**Lemma 2** *Let $\mathbf{H}$ and $\mathbf{H}^\dagger$ be as defined in the above construction. If $\mathbf{H}$ is an $\epsilon$-universal projective hash family, then $\mathbf{H}^\dagger$ is an $\epsilon$-universal$_2$ projective hash family.*

Proof. It is immediate that Definition 2 is satisfied.

The proof that Definition 3 is satisfied follows from a simple "conditioning argument," the details of which we now provide.

Consider the probability space defined by choosing $\vec{k} \in K^{2n}$ at random. To show that $\mathbf{H}^\dagger$ is $\epsilon$-universal$_2$, we have to show that for any $x, x^* \in X$ with $x \notin L \cup \{x^*\}$, conditioned on any fixed values of $H^\dagger_{\vec{k}}(x^*)$ and $\alpha^\dagger(\vec{k})$, the value of $H^\dagger_{\vec{k}}(x)$ can be guessed with probability at most $\epsilon$.

Let $\Gamma(x) = (\gamma_1, \ldots, \gamma_n) \in \{0,1\}^n$ and $\Gamma(x^*) = (\gamma_1^*, \ldots, \gamma_n^*) \in \{0,1\}^n$. Since $x \neq x^*$, we must have $\gamma_i \neq \gamma_i^*$ for some $1 \leq i \leq n$, and without loss of generality, let us assume that $i = n$.

In addition to conditioning on fixed values of $H^\dagger_{\vec{k}}(x^*)$ and $\alpha^\dagger(\vec{k})$, let us further condition on fixed values of $k_{1,0}, k_{1,1}, \ldots, k_{n-1,0}, k_{n-1,1}$, as well as $k_{n,\gamma_n^*}$ (consistent with the fixed values of $H^\dagger_{\vec{k}}(x^*)$ and $\alpha^\dagger(\vec{k})$). In this conditional probability space, the value of $H^\dagger_{\vec{k}}(x)$ determines the value of $H_{k_{n,\gamma_n}}(x)$, and thus, if the value of $H^\dagger_{\vec{k}}(x)$ could be guessed with probability greater than $\epsilon$, then so could the value of $H_{k_{n,\gamma_n}}(x)$. But since $\mathbf{H}$ is $\epsilon$-universal, it follows that the value of $H_{k_{n,\gamma_n}}(x)$ cannot be guessed with probability greater than $\epsilon$. We conclude that value of $H^\dagger_{\vec{k}}(x)$ cannot be guessed with probability greater than $\epsilon$ in this conditional probability space. Since this holds for all fixed values of $k_{1,0}, k_{1,1}, \ldots, k_{n-1,0}, k_{n-1,1}$, and $k_{n,\gamma_n^*}$ under consideration, it holds as well in the conditional probability space where just $H^\dagger_{\vec{k}}(x^*)$ and $\alpha^\dagger(\vec{k})$ are fixed, which proves the theorem. $\triangle$

The following construction is a variation on Lemma 2. It extends the sets $X$ and $L$ by taking the Cartesian product of these sets with a fixed, finite set $E$. Such extensions will prove useful in the sequel.

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be an $\epsilon$-universal projective hash family. Let $E$ be a non-empty, finite set.

Let us assume that we have injective functions $\Gamma : X \times E \to \{0,1\}^n$ and $\Gamma' : \Pi \to \{0,1\}^{n'}$ for some appropriately large positive integers $n$ and $n'$. Let $\mathbf{H}^\ddagger = (H^\ddagger, K^{2n}, X \times E, L \times E, \{0,1\}^{n'}, S^{2n}, \alpha^\ddagger)$, where $H^\ddagger$ and $\alpha^\ddagger$ are defined as follows.

For $\vec{k} = (k_{1,0}, k_{1,1}, \ldots, k_{n,0}, k_{n,1}) \in K^{2n}$, and $(x, e) \in X \times E$ with $\Gamma(x, e) = (\gamma_1, \ldots, \gamma_n) \in \{0,1\}^n$, we define

$$H^\ddagger_{\vec{k}}(x, e) = \bigoplus_{i=1}^{n} \Gamma'(H_{k_{i,\gamma_i}}(x))$$

and

$$\alpha^\ddagger(\vec{k}) = (\alpha(k_{1,0}), \alpha(k_{1,1}), \ldots, \alpha(k_{n,0}), \alpha(k_{n,1})).$$

The proof of the following lemma is essentially the same as the proof of Lemma 2.

**Lemma 3** *Let $\mathbf{H}$ and $\mathbf{H}^\ddagger$ be as defined in the above construction. If $\mathbf{H}$ is an $\epsilon$-universal projective hash family, then $\mathbf{H}^\ddagger$ is an $\epsilon$-universal$_2$ projective hash family.*

### 3.5.3 From universal projective to smooth projective

Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be an $\epsilon$-universal projective hash family. The next construction turns $\mathbf{H}$ into a $\delta$-smooth projective hash family $\mathbf{H}^*$ for $(X, L)$, where the hash outputs are $a$-bit strings, provided $\epsilon$ and $a$ are not too big, and $\delta$ is not too small.

The construction is a simple application of the Leftover Hash Lemma (a.k.a., Entropy Smoothing Lemma; see, e.g., [L, p. 86]).

Let $\mathbf{F} = (\check{H}, \check{K}, \Pi, \check{\Pi})$ be a pair-wise independent hash family, where $\check{\Pi} = \{0,1\}^a$ for some integer $a \geq 1$. Such a hash family can easily be constructed using well-known and quite practical techniques based on arithmetic in finite fields. We do not discuss this any further here.

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

**Lemma 4** *Let $\mathbf{H}$, $\mathbf{F}$, $\mathbf{H}^*$, and $a$ be as in the above construction. Suppose that $\mathbf{H}$ is an $\epsilon$-universal projective hash family. For any integer $b \geq 0$ such that $a + 2b \leq \log_2(1/\epsilon)$, $\mathbf{H}^*$ is a $2^{-(b+1)}$-smooth projective hash family.*

Proof. It is clear that $\mathbf{H}^*$ satisfies the basic requirements of a projective hash family.

Consider the random variables $U(\mathbf{H}^*)$ and $V(\mathbf{H}^*)$, as defined in the paragraph preceding Definition 4. That is, consider the probability space where $k \in K$, $\check{k} \in \check{K}$, $x \in X \setminus L$, and $\check{\pi}' \in \check{\Pi}$ are chosen at random, and set $U(\mathbf{H}^*) = (x, s, \check{k}, \check{\pi}')$ and $V(\mathbf{H}^*) = (x, s, \check{k}, \check{\pi})$, where $s = \alpha(k)$ and $\check{\pi} = \check{H}_{\check{k}}(H_k(x))$.

Consider any conditional probability space where particular values of $x \in X \setminus L$ and $s \in S$ are fixed, and let $U(\mathbf{H}^* \mid x, s)$ and $V(\mathbf{H}^* \mid x, s)$ 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 $\epsilon$-universal projective hashing, the distribution of $H_k(x)$ has min-entropy at least $\log_2(1/\epsilon)$, and $\check{k}$ is uniformly and independently distributed over $\check{K}$. The Leftover Hash Lemma then directly implies that $U(\mathbf{H}^* \mid x, s)$ and $V(\mathbf{H}^* \mid x, s)$ 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. $\triangle$

# 4 Subset membership problems

In this section we define a class of languages with some natural cryptographic indistinguishability properties. The definitions below capture the natural properties of well-known cryptographic problems such as the Quadratic Residuosity and Decision Diffie-Hellman problems, as well as others.

A *subset membership problem* $\mathbf{M}$ specifies a collection $(I_\ell)_{\ell \geq 0}$ of distributions. For every value of a security parameter $\ell \geq 0$, $I_\ell$ is a probability distribution of *instance descriptions.*

An instance description $\Lambda$ specifies the following:

- Finite, non-empty sets $X$, $L$, and $W$, such that $L$ is a proper subset of $X$.
- A binary relation $R \subset X \times W$.

For all $\ell \geq 0$, $[I_\ell]$ denotes the instance descriptions that are assigned non-zero probability in the distribution $I_\ell$. We write $\Lambda[X, L, W, R]$ to indicate that the instance $\Lambda$ specifies $X$, $L$, $W$ and $R$ as above.

For $x \in X$ and $w \in W$ with $(x, w) \in R$, we say that $w$ is a *witness* for $x$. Note that it would be quite natural to require that for all $x \in X$, we have $(x, w) \in R$ for some $w \in W$ if and only if $x \in L$, and that the relation $R$ is efficiently computable; however, we will not make these requirements here, as they are not necessary for our purposes. The actual role of a witness will become apparent in the next section.

A subset membership problem also provides several algorithms. For this purpose, we require that instance descriptions, as well as elements of the sets $X$ and $W$, can be uniquely encoded as bit strings of length polynomially bounded in $\ell$. The following algorithms are provided:

- a probabilistic, polynomial time sampling algorithm that on input $1^\ell$ for $\ell \geq 0$ samples an instance $\Lambda$ according to the distribution $I_\ell$.

  We do not require that the output distribution of the sampling algorithm and $I_\ell$ are equal; rather, we only require that they are $\iota(\ell)$-close, where $\iota(\ell)$ is a negligible function. In particular, with negligible probability, the sampling algorithm may output something that is not even an element of $[I_\ell]$.

  We call this algorithm the *instance sampling algorithm of* $\mathbf{M}$, and we call the statistical distance $\iota(\ell)$ discussed above its *approximation error.*

- a probabilistic, polynomial time sampling algorithm that takes as input $1^\ell$ for $\ell \geq 0$ and an instance $\Lambda[X, L, W, R] \in [I_\ell]$, and outputs a random $x \in L$, together with a witness $w \in W$ for $x$.

  We do not require that the distribution of the output value $x$ and the uniform distribution on $L$ are equal; rather, we only require that they are $\iota'(\ell)$-close, where $\iota'(\ell)$ is a negligible function. However, we do require that the output $x$ is always in $L$.

We call this algorithm the *subset sampling algorithm for* $\mathbf{M}$, and we call the statistical distance $\iota'(\ell)$ discussed above its *approximation error.*

- a deterministic, polynomial time algorithm that takes as input $1^\ell$ for $\ell \geq 0$, an instance $\Lambda[X, L, W, R] \in [I_\ell]$, and $\zeta \in \{0,1\}^*$, and checks whether $\zeta$ is a valid binary encoding of an element of $X$.

This completes the definition of a subset membership problem.

We next define the notion of a *hard* subset membership problem. Essentially, this means that it is computationally hard to distinguish random elements of $L$ from random elements of $X \setminus L$. We now formulate this notion more precisely.

Let $\mathbf{M}$ be a subset membership problem as above. We define two sequences of random variables, $(U_\ell(\mathbf{M}))_{\ell\geq 0}$ and $(V_\ell(\mathbf{M}))_{\ell\geq 0}$, as follows. Fix $\ell \geq 0$, and consider the probability space defined by sampling $\Lambda[X, L, W, R]$ from $I_\ell$, and choosing $x \in L$ at random and $x' \in X - L$ at random. Set $U_\ell(\mathbf{M}) = (\Lambda, x)$ and $V_\ell(\mathbf{M}) = (\Lambda, x')$.

**Definition 6** *Let* $\mathbf{M}$ *be a subset membership problem. We say that* $\mathbf{M}$ *is* hard *if* $(U_\ell(\mathbf{M}))_{\ell\geq 0}$ *and* $(V_\ell(\mathbf{M}))_{\ell\geq 0}$ *are computationally indistinguishable.*

# 5 Universal hash proof systems

## 5.1 Hash proof systems

Let $\mathbf{M}$ be a subset membership problem, as defined in §4, specifying a sequence $(I_\ell)_{\ell\geq 0}$ of instance distributions.

A *hash proof system (HPS)* $\mathbf{P}$ for $\mathbf{M}$ associates with each instance $\Lambda[X, L, W, R]$ of $\mathbf{M}$ a projective hash family $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ for $(X, L)$.

Additionally, $\mathbf{P}$ provides several algorithms to carry out basic operations we have defined for an associated projective hash family; namely, sampling $k \in K$ at random, computing $\alpha(k) \in S$ given $k \in K$, computing $H_k(x) \in \Pi$ given $k \in K$ and $x \in X$. We call this latter algorithm the *private evaluation algorithm for* $\mathbf{P}$. Moreover, a crucial property is that the system provides an efficient algorithm to compute $H_k(x) \in \Pi$, given $\alpha(k) \in S$, $x \in L$, and $w \in W$, where $w$ is a witness for $x$. We call this algorithm the *public evaluation algorithm for* $\mathbf{P}$. The system should also provide an algorithm that recognizes elements of $\Pi$.

We now discuss the above-mentioned algorithms in a bit more detail. In this discussion, whenever $\Lambda[X, L, W, R] \in [I_\ell]$ is fixed in some context, it is to be understood that $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ is the projective hash family that $\mathbf{P}$ associates with $\Lambda$. These algorithms work with bit strings of length bounded by a polynomial in $\ell$ to represent elements of $K$, $\Pi$ and $S$. We also assume that these algorithms use the same encodings of the sets $X$, $L$ and $W$ as the algorithms from the subset membership problem $\mathbf{M}$.

The system $\mathbf{P}$ provides the following algorithms:

- a probabilistic, polynomial time algorithm that takes as input $1^\ell$ and an instance $\Lambda \in [I_\ell]$, and outputs $k \in K$, distributed uniformly over $K$.

- a deterministic, polynomial time algorithm that takes as input $1^\ell$, an instance $\Lambda \in [I_\ell]$, $k \in K$, and outputs $s \in S$ such that $\alpha(k) = s$.

- a deterministic, polynomial time algorithm that takes as input $1^\ell$, an instance $\Lambda \in [I_\ell]$, $k \in K$ and $x \in X$, and outputs $\pi \in \Pi$ such that $H_k(x) = \pi$.

  This is the *private evaluation algorithm.*

- a deterministic, polynomial time algorithm that takes as input $1^\ell$, an instance $\Lambda \in [I_\ell]$, $s \in S$ such that $\alpha(k) = s$ for some $k \in S$, and $x \in L$ together with a witness $w \in W$ for $x$, and outputs $\pi \in \Pi$ such that $H_k(x) = \pi$.

  This is the *public evaluation algorithm.*

- a deterministic, polynomial time algorithm that takes as input $1^\ell$, an instance $\Lambda \in [I_\ell]$, and $\zeta \in \{0,1\}^*$, and determines if $\zeta$ is a valid encoding of an element of $\Pi$.

### 5.2 Universal hash proof systems

**Definition 7** *Let $\epsilon(\ell)$ be a function mapping non-negative integers to non-negative reals. Let* $\mathbf{M}$ *be a subset membership problem specifying a sequence $(I_\ell)_{\ell \geq 0}$ of instance distributions. Let* $\mathbf{P}$ *be an HPS for* $\mathbf{M}$.

*We say that* $\mathbf{P}$ *is* $\epsilon(\ell)$-universal (respectively, -universal$_2$, -smooth) *if there exists a negligible function $\delta(\ell)$ such that for all $\ell \geq 0$ and for all $\Lambda[X, L, W, R] \in [I_\ell]$, the projective hash family $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ that* $\mathbf{P}$ *associates with $\Lambda$ is $\delta(\ell)$-close to an $\epsilon(\ell)$-universal (respectively, -universal$_2$, -smooth) projective hash family $\mathbf{H}^* = (H^*, K^*, X, L, \Pi, S, \alpha^*)$.*

*Moreover, if this is the case, and $\epsilon(\ell)$ is a negligible function, then we say that* $\mathbf{P}$ *is* strongly universal (respectively, universal$_2$, smooth).

We shall call the function $\delta(\ell)$ in the above definition the *approximation error* of $\mathbf{P}$, and we shall refer to the projective hash family $\mathbf{H}^*$ as the *idealization* of $\mathbf{H}$.

It is perhaps worth remarking that if a hash proof system is strongly universal, and the underlying subset membership problem is hard, then the problem of evaluating $H_k(x)$ for random $k \in K$ and arbitrary $x \in X$, given only $x$ and $\alpha(k)$, must be hard.

We also need an extension of this notion.

The definition of an *extended* HPS $\mathbf{P}$ for $\mathbf{M}$ is the same as that of ordinary HPS for $\mathbf{M}$, except that for each $k \geq 0$ and for each $\Lambda = \Lambda[X, L, W, R] \in [I_k]$, the proof system $\mathbf{P}$ associates with $\Lambda$ a finite set $E$ along with a projective hash family $\mathbf{H} = (H, K, X \times E, L \times E, \Pi, S, \alpha)$ for $(X \times E, L \times E)$. Note that in this setting, to compute $H_k(x, e)$ for $x \in L$ and $e \in E$, the public evaluation algorithm takes as input $\alpha(k) \in S$, $x \in L$, $e \in E$, and a witness $w \in W$ for $x$, and the private evaluation algorithm takes as input $k \in K$, $x \in X$, and $e \in E$. We shall also require that elements of $E$ are uniquely encoded as bit strings of length bounded by a polynomial in $\ell$, and that $\mathbf{P}$ provides an algorithm that efficiently determines whether a bit string is a valid encoding of an element of $E$.

Definition 7 can be modified in the obvious way to define *extended $\epsilon(\ell)$-universal$_2$ HPS's* (we do not need any of the other notions, nor are they particularly interesting).

#### 5.2.1 Constructions

Note that based on the constructions in Lemmas 1, 2, 3, and 4, given an HPS that is (say) 1/2-universal, we can construct a strongly universal HPS, a (possibly extended) strongly universal$_2$ HPS, and a strongly smooth HPS. However, in most special cases of practical interest, there are much more efficient constructions.

# 6 A general framework for 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.

## 6.1 Public-key encryption schemes

A public key encryption scheme provides three algorithms:

- a probabilistic, polynomial-time *key generation* algorithm that on input $1^\ell$, where $\ell \geq 0$ is a security parameter, outputs a public-key/private-key pair $(\mathsf{PK}, \mathsf{SK})$.

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

- a probabilistic, polynomial-time *encryption* algorithm that on input $1^\ell$, $\mathsf{PK}$, and $m$, where $\ell \geq 0$, $\mathsf{PK}$ is a public key associated with security parameter $\ell$, and $m \in M_{\mathsf{PK}}$, outputs a bit string $\sigma$.

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

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 $(\mathsf{PK}, \mathsf{SK})$ *bad* if for some $m \in M_{\mathsf{PK}}$, and for some encryption $\sigma$ of $m$ under $\mathsf{PK}$, the decryption of $\sigma$ under $\mathsf{SK}$ is not $m$. Let us call a public-key encryption scheme *sound* if the probability that the key generation algorithm on input $1^\ell$ outputs a bad key pair is a negligible function in $\ell$.

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.

Note that in this paper, we only work with finite message spaces.

## 6.2 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 $\ell \geq 0$ be the security parameter. We run the key-generation algorithm of the public-key encryption scheme on input $1^\ell$, and get a key pair $(\mathsf{PK}, \mathsf{SK})$.

   We equip an *encryption oracle* with the public key $\mathsf{PK}$, and a *decryption oracle* with the secret key $\mathsf{SK}$.

   The public-key $\mathsf{PK}$ 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* $\sigma$ 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 $\mathsf{SK}$ and the query $\sigma$, and responds to the query by returning the output to the adversary.

   Note that a query is not required to represent an encryption (under $\mathsf{PK}$) of a message; a query can indeed be any string designed to probe the behavior 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 $m_\beta$ under $\mathsf{PK}$.

   The resulting encryption $\sigma^*$, 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 $\sigma$ that are different from the target ciphertext $\sigma^*$.

5. *Guessing-Phase.* The adversary outputs a bit $\hat{\beta}$.

The adversary is said to *win* the game if $\hat{\beta} = \beta$. We define the *advantage* (over random guessing) of the adversary as the absolute value of the difference of the probability that he wins and 1/2.

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.

## 6.3 The generic scheme and its analysis

We now describe our generic method for constructing a secure public-key encryption scheme.

Let $\mathbf{M}$ be a subset membership problem specifying a sequence $(I_\ell)_{\ell \geq 0}$ of instance distributions. We also need a strongly smooth hash proof system $\mathbf{P}$ for $\mathbf{M}$, as well as a strongly universal$_2$ extended hash proof system $\hat{\mathbf{P}}$ for $\mathbf{M}$. We discuss $\mathbf{P}$ and $\hat{\mathbf{P}}$ below in greater detail.

To simplify the notation, we will describe the scheme with respect to a fixed value $\ell \geq 0$ of the security parameter, and a fixed instance description $\Lambda[X, L, W, R] \in [I_\ell]$. 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 $\mathbf{M}$, 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 $\Lambda$ fixed as above, let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be the projective hash family that $\mathbf{P}$ associates with $\Lambda$, and let $\hat{\mathbf{H}} = (\hat{H}, \hat{K}, X \times \Pi, L \times \Pi, \hat{\Pi}, \hat{S}, \hat{\alpha})$ be the projective hash family that $\hat{\mathbf{P}}$ associates with $\Lambda$. We require that $\Pi$ is an abelian group, for which we use additive notation, and that elements of $\Pi$ 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 $\Lambda$, with corresponding projective hash families $\mathbf{H}$ and $\hat{\mathbf{H}}$, as above. The message space is $\Pi$.

**Key Generation**

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

The public key is $(s, \hat{s})$.

The private key is $(k, \hat{k})$.

**Encryption**

To encrypt a message $m \in \Pi$ 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 $\mathbf{M}$.

Compute $\pi = H_k(x) \in \Pi$, using the public evaluation algorithm for $\mathbf{P}$ on inputs $s$, $x$, and $w$.

Compute $e = m + \pi \in \Pi$.

Compute $\hat{\pi} = \hat{H}_{\hat{k}}(x, e) \in \hat{\Pi}$, using the public evaluation algorithm for $\hat{\mathbf{P}}$ on inputs $\hat{s}$, $x$, $e$, and $w$.

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$ under a secret key as above, one does the following.

Compute $\hat{\pi}' = \hat{H}_{\hat{k}}(x, e) \in \hat{\Pi}$, using the private evaluation algorithm for $\hat{\mathbf{P}}$ on inputs $\hat{k}$, $x$, and $e$.

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute $\pi = H_k(x) \in \Pi$, using the private evaluation algorithm for $\mathbf{P}$ on inputs $k$ and $x$.

Compute $m = e - \pi \in \Pi$, 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 $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$; if not, the decryption algorithm outputs reject and halts.

We remark that to implement this scheme, all we really need is a 1/2-universal HPS, since we can convert this into appropriate strongly smooth and strongly universal$_2$ HPS's using the general constructions discussed in §5.2.1. Indeed, the Leftover Hash construction in Lemma 4 gives us a strongly smooth HPS whose hash outputs are bit strings of a given length $a$, and so we can take the group $\Pi$ in the above construction to be the group of $a$-bit strings with "exclusive or" as the group operation.

**Theorem 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 $1^\ell$, for $\ell \geq 0$, along with $\Lambda[X, L, W, R] \in [I_\ell]$, and $x^* \in X$.

The simulator provides a "simulated environment" for the adversary as follows. In this description, $\mathbf{H}$ and $\hat{\mathbf{H}}$ 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 $\Lambda$.

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 $\beta$, and computes the target ciphertext $(x^*, e^*, \hat{\pi}^*)$, 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 $\mathbf{P}$ on inputs $k$ and $x^*$. It then computes $e^* = m_\beta + \pi^*$. Finally, it computes $\hat{\pi}^* = \hat{H}_{\hat{k}}(x^*, e^*)$, using the *private* evaluation algorithm for $\hat{\mathbf{P}}$ on inputs $\hat{k}$, $x^*$, and $e^*$.

In the Guessing Phase, the adversary outputs a bit $\hat{\beta}$. The simulator outputs 1 if $\beta = \hat{\beta}$, and 0 otherwise, after which, the simulator halts.

For each value of the security parameter $\ell \geq 0$, we consider the behavior of this simulator/adversary pair in two different experiments. In the first experiment, the simulator is given $(\Lambda, x^*)$, where $\Lambda[X, L, W, R]$ is sampled from $I_\ell$, and $x^*$ is sampled at random from $L$; let $T'_\ell$ be the event that the simulator outputs a 1 in this experiment. In the second experiment, the simulator is given $(\Lambda, x^*)$, where $\Lambda[X, L, W, R]$ is sampled from $I_\ell$, and $x^*$ is sampled at random from $X \setminus L$; let $T_\ell$ be the event that the simulator outputs a 1 in this experiment.

Let $\mathsf{AdvDist}(\ell) = |\Pr[T_\ell] - \Pr[T'_\ell]|$; that is, $\mathsf{AdvDist}(\ell)$ is the distinguishing advantage of our simulator. Let $\mathsf{AdvCCA}(\ell)$ be the adversary's advantage in an adaptive chosen ciphertext attack. Our goal is to show that $\mathsf{AdvCCA}(\ell)$ is negligible, provided $\mathsf{AdvDist}(\ell)$ 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 $\mathbf{P}$ is $\epsilon(\ell)$-smooth with approximation error $\delta(\ell)$, and that $\hat{\mathbf{P}}$ is $\hat{\epsilon}(\ell)$-universal$_2$ with approximation error $\hat{\delta}(\ell)$. Also, we assume that the instance sampling algorithm for $\mathbf{M}$ has approximation error $\iota(\ell)$, and that the subset sampling algorithm for $\mathbf{M}$ has approximation error $\iota'(\ell)$.

**Case $x^* \in L$.** In this case, the simulation is perfect, except for the approximation errors introduced by the instance and subset sampling algorithms for $\mathbf{M}$. Thus, we have

$$|\Pr[T'_\ell] - 1/2| \geq \mathsf{AdvCCA}(\ell) - (\iota(\ell) + \iota'(\ell)). \tag{1}$$

**Case $x^* \in X \setminus L$.** To analyze the behavior 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 1, 2, etc. Each of these experiments are best viewed as operating on the same underlying probability space; we define the event $T_\ell^{(i)}$, 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 1.* To define experiment 1, we modify the simulator as follows. We replace the projective hash family $\mathbf{H}$ that $\mathbf{P}$ associates with $\Lambda$ with its idealization, which is an $\epsilon(\ell)$-smooth projective hash family that is $\delta(\ell)$-close to $\mathbf{H}$. We also replace the projective hash family $\hat{\mathbf{H}}$ that $\hat{\mathbf{P}}$ associates with $\Lambda$ with its idealization, which is an $\hat{\epsilon}(\ell)$-universal$_2$ projective hash family that is $\hat{\delta}(\ell)$-close to $\hat{\mathbf{H}}$. By definition, we have

$$|\Pr[T_\ell^{(1)}] - \Pr[T_\ell^{(0)}]| \leq \delta(\ell) + \hat{\delta}(\ell). \tag{2}$$

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

*Experiment 2.* In experiment 2, we modify the simulator yet again, so that in addition to rejecting a ciphertext $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$ if $\hat{H}_{\hat{k}}(x, e) \neq \hat{\pi}$, the decryption oracle also rejects the ciphertext if $x \notin L$. Let $F_2$ be the event in experiment 2 that some ciphertext $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$ with $x \notin L$ is rejected by the decryption oracle but $\hat{H}_{\hat{k}}(x, e) = \hat{\pi}$.

We claim that

$$\Pr[F_2] \leq Q(\ell)\hat{\epsilon}(\ell). \tag{3}$$

To prove (3), let us condition on a fixed value of $\Lambda[X, L, W, R]$ (which determines the projective hash families $\mathbf{H}$ and $\hat{\mathbf{H}}$), as well as fixed values of $k$, $\hat{s}$, 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 $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$, with $x \notin L$, that is submitted as a decryption oracle query during Probing Phase I. In this conditional probability space, $x$, $e$, and $\hat{\pi}$ are fixed, whereas $\hat{k}$ is still uniformly distributed over $\hat{K}$, subject only to the constraint that $\hat{\alpha}(\hat{k}) = \hat{s}$, where $\hat{s}$ is fixed as above. Therefore, from the $\hat{\epsilon}(\ell)$-universal$_2$ property of $\hat{\mathbf{H}}$, the probability that $\hat{H}_{\hat{k}}(x, e) = \hat{\pi}$ in this conditional probability space is at most $\hat{\epsilon}(\ell)$.

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

- Suppose that $(x, e) = (x^*, e^*)$. Since we must have $(x, e, \hat{\pi}) \neq (x^*, e^*, \hat{\pi}^*)$, it follows that $\hat{\pi} \neq \hat{\pi}^*$, and hence $\hat{H}_{\hat{k}}(x, e) \neq \hat{\pi}$ with certainty.

- Suppose that $(x, e) \neq (x^*, e^*)$. In this conditional probability space, $x$, $e$, and $\hat{\pi}$ are fixed, whereas $\hat{k}$ is still uniformly distributed over $\hat{K}$, subject only to the constraint that $\hat{\alpha}(\hat{k}) = \hat{s}$ and $\hat{H}_{\hat{k}}(x^*, e^*) = \hat{\pi}^*$, where $\hat{s}$, $x^*$, $e^*$, and $\hat{\pi}^*$ are fixed as above. Therefore, from the $\hat{\epsilon}(\ell)$-universal$_2$ property of $\hat{\mathbf{H}}$, the probability that $\hat{H}_{\hat{k}}(x, e) = \hat{\pi}$ in this conditional probability space is at most $\hat{\epsilon}(\ell)$.

The above arguments show that for any individual ciphertext $(x, e, \hat{\pi}) \in X \times \Pi \times \hat{\Pi}$, with $x \notin L$, that is submitted to the decryption oracle, the probability that $\hat{H}_{\hat{k}}(x, e) = \hat{\pi}$ is at most $\hat{\epsilon}(\ell)$, from which the bound (3) immediately follows.

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

$$|\Pr[T_\ell^{(2)}] - \Pr[T_\ell^{(1)}]| \leq \Pr[F_2]. \tag{4}$$

*Experiment 3.* In experiment 3, we modify the simulator yet again. This time, in the encryption oracle, instead of computing $\pi^*$ as $H_k(x^*)$, the simulator sets $\pi^* = \pi'$, where $\pi' \in \Pi$ is chosen at random. Now, let us condition on a fixed value of $\Lambda[X, L, W, R]$ (which determines the projective hash families $\mathbf{H}$ and $\hat{\mathbf{H}}$), as well as fixed values of $\hat{k}$, $\beta$, 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 $(x, e, \hat{\pi})$ with $x \notin L$, it follows that the output of the simulator in experiment 2 is completely determined as a function of $x^*$, $s$, and $H_k(x^*)$, while the output in experiment 3 is determined as the *same* function of $x^*$, $s$, and $\pi'$. Moreover, by independence, the joint distribution of $(k, x^*, \pi')$ does not change in passing from the original probability space to the conditional probability space. It now follows directly from the $\epsilon(\ell)$-smooth property of $\mathbf{H}$ that

$$|\Pr[T_\ell^{(3)}] - \Pr[T_\ell^{(2)}]| \le \epsilon(\ell). \tag{5}$$

It is evident from the definition of the simulator in experiment 3 that the adversary's output $\hat{\beta}$ in this experiment is independent of the hidden bit $\beta$; therefore,

$$\Pr[T_\ell^{(3)}] = 1/2. \tag{6}$$

*Putting it all together.* Combining the relations (2)-(6), we see that

$$|\Pr[T_\ell] - 1/2| \le \delta(\ell) + \epsilon(\ell) + \hat{\delta}(\ell) + Q(\ell)\hat{\epsilon}(\ell). \tag{7}$$

Combining the inequalities (1) and (7), we see that

$$\mathsf{AdvCCA}(\ell) \le \mathsf{AdvDist}(\ell) + \delta(\ell) + \epsilon(\ell) + \hat{\delta}(\ell) + Q(\ell)\hat{\epsilon}(\ell) + \iota(\ell) + \iota'(\ell), \tag{8}$$

from which the theorem immediately follows. △

# 7 Universal projective hash families: constructions

We now present group-theoretic constructions of universal projective hash families.

## 7.1 Diverse group systems and derived projective hash families

Let $X$, $L$ and $\Pi$ be finite abelian groups, where $L$ is a proper subgroup of $X$. We will use additive notation for these groups.

Let $\mathrm{Hom}(X, \Pi)$ denote the group of all homomorphisms $\phi : X \to \Pi$. This is also a finite abelian group for which we use additive notation as well. For $\phi, \phi' \in \mathrm{Hom}(X, \Pi)$, $x \in X$, and $a \in \mathbb{Z}$, we have $(\phi + \phi')(x) = \phi(x) + \phi'(x)$, $(\phi - \phi')(x) = \phi(x) - \phi'(x)$, and $(a\phi)(x) = a\phi(x) = \phi(ax)$. The zero element of $\mathrm{Hom}(X, \Pi)$ sends all elements of $X$ to $0 \in \Pi$.

**Definition 8** *Let $X, L, \Pi$ be as above. Let $\mathcal{H}$ be a subgroup of $Hom(X, \Pi)$. We call $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ a* group system.

Let $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ be a group system, and let $g_1, \ldots, g_d \in L$ be a set of generators for $L$. Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$, where

- for randomly chosen $k \in K$, $H_k$ is uniformly distributed over $\mathcal{H}$,
- $S = \Pi^d$, and

- the map $\alpha : K \to S$ sends $k \in K$ to $(\phi(g_1), \ldots, \phi(g_d)) \in S$, where $\phi = H_k$.

It is easily seen that **H** is a projective hash family. To see this, note that if $x \in L$, then there exist $w_1, \ldots, w_d \in \mathbb{Z}$ such that $x = \sum_{i=1}^{d} w_i g_i$; now, for $k \in K$ with $H_k = \phi$ and $\alpha(k) = (\mu_1, \ldots, \mu_d)$, we have

$$H_k(x) = \phi(\sum_{i=1}^{d} w_i g_i) = \sum_{i=1}^{d} w_i \phi(g_i) = \sum_{i=1}^{d} w_i \mu_i.$$

Thus, the action of $H_k$ on $L$ is determined by $\alpha(k)$, as required.

**Definition 9** *Let* **G** *be a group system as above and let* **H** *be a projective hash family as above. Then we say that* **H** *is a projective hash family* derived from **G**.

Looking ahead, we remark that the reason for defining $\alpha$ in this way is to facilitate efficient implementation of the public evaluation algorithm for a hash proof system with which **H** may be associated. In this context, if a "witness" for $x$ is $(w_1, \ldots, w_d)$ as above, then $H_k(x)$ can be efficiently computed from $\alpha(k)$ and $(w_1, \ldots, w_d)$, assuming arithmetic in $\Pi$ is efficiently implemented.

Our first goal is to investigate the conditions under which a projective hash family derived from a group system is $\epsilon$-universal for some $\epsilon < 1$.

**Definition 10** *Let* $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ *be a group system. We say that* **G** *is* diverse *if for all* $x \in X \setminus L$*, there exists* $\phi \in \mathcal{H}$ *such that* $\phi(L) = \langle 0 \rangle$*, but* $\phi(x) \neq 0$*.*

It is not difficult to see that diversity is a necessary condition for a group system if any derived projective hash family is to be $\epsilon$-universal for some $\epsilon < 1$. We will show in Theorem 2 below that any projective hash family derived from a diverse group system is $\epsilon$-universal, where $\epsilon = 1/\tilde{p}$, and $\tilde{p}$ is the smallest prime dividing $|X/L|$.

## 7.2 A universal projective hash family

Throughout this section, $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ denotes a group system, $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ denotes a projective hash family derived from **G**, and $\tilde{p}$ denotes the smallest prime dividing $|X/L|$.

**Definition 11** *For a set* $Y \subset X$*, let us define* $\mathcal{A}(Y)$ *to be the set of* $\phi \in \mathcal{H}$ *such that* $\phi(x) = 0$ *for all* $x \in Y$*; that is,* $\mathcal{A}(Y)$ *is the collection of homomorphisms in* $\mathcal{H}$ *that annihilate* $Y$*.*

It is clear that $\mathcal{A}(Y)$ is a subgroup of $\mathcal{H}$, and that $\mathcal{A}(Y) = \mathcal{A}(\langle Y \rangle)$.

The following is a straightforward re-statement of Definition 10.

**Lemma 5** **G** *is diverse if and only if for all* $x \in X \setminus L$*,* $\mathcal{A}(L \cup \{x\})$ *is a proper subgroup of* $\mathcal{A}(L)$*.*

**Lemma 6** *If* $p$ *is a prime dividing* $|\mathcal{A}(L)|$*, then* $p$ *divides* $|X/L|$*.*

Proof. Let $p$ be a prime dividing $|\mathcal{A}(L)|$. Then there exists an element $\phi \in \mathcal{A}(L)$ of order $p$. Let $a = |X/L|$, and note that for all $x \in X$, we must have $ax \in L$, since $a$ is the order of the factor group $X/L$. Therefore, for all $x \in X$, we have $(a \cdot \phi)(x) = \phi(ax) = 0$, the latter equality holding since $\phi$ annihilates $L$ and $ax \in L$. It follows that $p$ divides $a$. △

**Definition 12** *For* $x \in X$*, let* $\mathcal{E}_x : \mathcal{H} \to \Pi$ *be the map that sends* $\phi \in \mathcal{H}$ *to* $\phi(x) \in \Pi$*. Let us also define* $\mathcal{I}(x) = \mathcal{E}_x(\mathcal{A}(L))$*.*

Clearly, $\mathcal{E}_x$ is a group homomorphism, and $\mathcal{I}(x)$ is a subgroup of $\Pi$.

**Lemma 7** *If* $\mathbf{G}$ *is diverse, then for all* $x \in X \setminus L$, $|\mathcal{I}(x)|$ *is at least* $\tilde{p}$.

PROOF. Let $x \in X \setminus L$. Consider the restriction of the map $\mathcal{E}_x$ to $\mathcal{A}(L)$. The image of this map is $\mathcal{I}(x)$, and the kernel is $\mathcal{A}(L \cup \{x\})$. Therefore, $\mathcal{I}(x)$ is isomorphic to the factor group $\mathcal{A}(L)/\mathcal{A}(L \cup \{x\})$. Since $\mathbf{G}$ is assumed diverse, by Lemma 5, $\mathcal{A}(L \cup \{x\})$ is a proper subgroup of $\mathcal{A}(L)$. Thus, the order order of $\mathcal{I}(x)$ is a divisor of $\mathcal{A}(L)$ not equal to 1, and so is divisible by some prime $p$ dividing $\mathcal{A}(L)$. By Lemma 6, this prime $p$ divides $|X/L|$. $\triangle$

**Lemma 8** *Let* $s \in \alpha(K)$ *be fixed. Consider the probability space defined by choosing* $k \in \alpha^{-1}(s)$ *at random, and let* $\rho = H_k$. *Then* $\rho$ *is uniformly distributed over a coset* $\psi_s + \mathcal{A}(L)$ *of* $\mathcal{A}(L)$ *in* $\mathcal{H}$, *the precise coset depending on* $s$.

PROOF. Let $g_1, \ldots, g_d$ be the set of generators defining $\alpha$. Let $\tilde{\alpha} : \mathcal{H} \to S$ be the map that sends $\phi \in \mathcal{H}$ to $(\phi(g_1), \ldots, \phi(g_d)) \in S$. It is evident that $\rho$ is uniformly distributed over $\tilde{\alpha}^{-1}(s)$. Moreover, $\tilde{\alpha}$ is clearly a group homomorphism with kernel $\mathcal{A}(\{g_1, \ldots, g_d\}) = \mathcal{A}(L)$. It follows that $\tilde{\alpha}^{-1}(s)$ is a coset of $\mathcal{A}(L)$ in $\mathcal{H}$. $\triangle$

In Lemma 8, there are many choices for the "coset leader" $\psi_s \in \mathcal{H}$; however, let us fix one such choice arbitrarily, so that for the for the rest of this section $\psi_s$ denotes this coset leader.

**Theorem 2** *Let* $s \in \alpha(K)$ *and* $x \in X$ *be fixed. Consider the probability space defined by choosing* $k \in \alpha^{-1}(s)$ *at random, and let* $\pi = H_k(x)$. *Then* $\pi$ *is uniformly distributed over a coset of* $\mathcal{I}(x)$ *in* $\Pi$ *(the precise coset depending on* $s$ *and* $x$*). In particular, if* $\mathbf{G}$ *is diverse, then* $\mathbf{H}$ *is* $1/\tilde{p}$*-universal.*

PROOF. Let $\rho = H_k$. By Lemma 8, $\rho$ is uniformly distributed over $\psi_s + \mathcal{A}(L)$. Since $\pi = \rho(x)$, it follows that $\pi$ is uniformly distributed over $\mathcal{E}_x(\psi_s + \mathcal{A}(L)) = \psi_s(x) + \mathcal{I}(x)$. That proves the first statement of the theorem. The second statement follows immediately from Lemma 7, and the fact that $|\psi_s(x) + \mathcal{I}(x)| = |\mathcal{I}(x)|$. $\triangle$

### 7.3 A universal$_2$ projective hash family

We continue with the notation established in §7.2; in particular, $\mathbf{G} = (\mathcal{H}, X, L, \Pi)$ denotes a group system, $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ denotes a projective hash family derived from $\mathbf{G}$, and $\tilde{p}$ denotes the smallest prime dividing $|X/L|$.

Starting with $\mathbf{H}$, and applying the construction of Lemma 2 or Lemma 3, we can obtain a universal$_2$ projective hash family. However, by exploiting the group structure underlying $\mathbf{H}$, we can construct a more efficient universal$_2$ projective hash family $\hat{\mathbf{H}}$.

Let $E$ be an arbitrary finite set. $\hat{\mathbf{H}}$ is to be a projective hash family for $(X \times E, L \times E)$. Fix an injective encoding function

$$\Gamma : X \times E \to \{0, \ldots, \tilde{p} - 1\}^n,$$

where $n$ is sufficiently large.

Let $\hat{\mathbf{H}} = (\hat{H}, K^{n+1}, X \times E, L \times E, \Pi, S^{n+1}, \hat{\alpha})$, where $\hat{H}$ and $\hat{\alpha}$ are defined as follows. For $\vec{k} = (k', k_1, \ldots, k_n) \in K^{n+1}$, $x \in X$, and $e \in E$, we define

$$\hat{H}_{\vec{k}}(x, e) = H_{k'}(x) + \sum_{i=1}^{n} \gamma_i H_{k_i}(x),$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e)$, and we define

$$\hat{\alpha}(\vec{k}) = (\alpha(k'), \alpha(k_1), \ldots, \alpha(k_n)).$$

It is clear that $\hat{\mathbf{H}}$ is a projective hash family. We shall prove:

**Theorem 3** *Let $\hat{\mathbf{H}}$ be as above. Let $\vec{s} \in \alpha(K)^{n+1}$, $x, x^* \in X$, and $e, e^* \in E$ be fixed, where $(x, e) \neq (x^*, e^*)$. Consider the probability space defined by choosing $\vec{k} \in \hat{\alpha}^{-1}(\vec{s})$ at random, and let $\pi = \hat{H}_{\vec{k}}(x, e)$ and $\pi^* = \hat{H}_{\vec{k}}(x^*, e^*)$. Then $\pi$ is uniformly distributed over a coset of $\mathcal{I}(x)$ in $\Pi$ (the precise coset depending on $s$, $x$, and $e$), and $\pi^*$ is uniformly and independently distributed over a coset of $\mathcal{I}(x^*)$ in $\Pi$ (the precise coset depending on $s$, $x^*$, and $e^*$). In particular, if the underlying group system $\mathbf{G}$ is diverse, then $\hat{\mathbf{H}}$ is $1/\tilde{p}$-universal$_2$.*

Before proving this theorem, we state another elementary lemma.

Let $M \in \mathbb{Z}^{a \times b}$ be an integer matrix with $a$ rows and $b$ columns. Let $\mathcal{G}$ be a finite abelian group. Let $\mathbf{T}(M, \mathcal{G}) : \mathcal{G}^b \to \mathcal{G}^a$ be the map that sends $\vec{u} \in \mathcal{G}^b$ to $\vec{v} \in \mathcal{G}^a$, where

$$\vec{v}^\top = M \vec{u}^\top;$$

here, $(\cdots)^\top$ denotes transposition. Clearly, $\mathbf{T}(M, \mathcal{G})$ is a group homomorphism.

**Lemma 9** *Let $M$ and $\mathcal{G}$ be as above. If for all primes $p$ dividing $|\mathcal{G}|$, the rows of $M$ are linearly independent modulo $p$, then $\mathbf{T}(M, \mathcal{G})$ is surjective.*

Proof. The proof is by basic linear algebra, and we include it for completeness. Let $\prod_{i=1}^r p_i^{c_i}$ be the prime factorization of $|\mathcal{G}|$. From the conditions of the lemma, it follows that for each $1 \leq i \leq r$, there is a square sub-matrix $M_i$, consisting of $a$ columns of $M$, that is invertible over $\mathbb{Z}_{p_i}$ and, therefore, also over $\mathbb{Z}_{p_i^{c_i}}$. Hence, for each $1 \leq i \leq r$ there is a matrix $N_i \in \mathbb{Z}^{b \times a}$ such that $M \cdot N_i \equiv I \pmod{p_i^{c_i}}$, where $I$ is the $a \times a$ identity matrix over $\mathbb{Z}$. Combining $N_1, \ldots, N_r$ using the Chinese Remainder Theorem, there is a matrix $N \in \mathbb{Z}^{b \times a}$ such that $M \cdot N \equiv I \pmod{|\mathcal{G}|}$. Hence, for all $\vec{v} \in \mathcal{G}^a$, we have $\vec{v}^\top = M \vec{u}^\top$, where $\vec{u}^\top = N \vec{v}^\top$. $\triangle$

**Proof of Theorem 3.** Let $\vec{s} = (s', s_1, \ldots, s_n)$, $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e)$, and $(\gamma_1^*, \ldots, \gamma_n^*) = \Gamma(x^*, e^*)$. Let $(\rho', \rho_1, \ldots, \rho_n) = (H_{k'}, H_{k_1}, \ldots, H_{k_n})$.

Now define the matrix $M \in \mathbb{Z}^{2 \times (n+1)}$ as

$$M = \begin{pmatrix} 1 & \gamma_1 & \gamma_2 & \cdots & \gamma_n \\ 1 & \gamma_1^* & \gamma_2^* & \cdots & \gamma_n^* \end{pmatrix},$$

so that if

$$(\tilde{\rho}, \tilde{\rho}^*)^\top = M(\rho', \rho_1, \ldots, \rho_n)^\top$$

then we have $(\pi, \pi^*) = (\rho(x), \rho^*(x^*))$.

By the definition of $\Gamma$, and by Lemma 6, we see that $(\gamma_1, \ldots, \gamma_n)$ and $(\gamma_1^*, \ldots, \gamma_n^*)$ are distinct modulo any prime $p$ that divides $\mathcal{A}(L)$. Therefore, Lemma 9 implies that the map $\mathbf{T}(M, \mathcal{A}(L))$ is surjective. By Lemma 8, $(\rho', \rho_1, \ldots, \rho_n)$ is uniformly distributed over

$$(\psi_{s'} + \mathcal{A}(L), \psi_{s_1} + \mathcal{A}(L), \ldots, \psi_{s_n} + \mathcal{A}(L)).$$

Thus, $(\tilde{\rho}, \tilde{\rho}^*)$ is uniformly distributed over $(\tilde{\psi} + \mathcal{A}(I), \tilde{\psi}^* + \mathcal{A}(I))$, where

$$(\tilde{\psi}, \tilde{\psi}^*)^\top = M(\psi_{s'}, \psi_{s_1}, \ldots, \psi_{s_n})^\top.$$

It follows that $(\pi, \pi^*)$ is uniformly distributed over $(\tilde{\psi}(x) + \mathcal{I}(x), \tilde{\psi}^*(x^*) + \mathcal{I}(x^*))$.

That proves the first statement of the theorem. The second statement now follows from Lemma 7. $\triangle$

If $\tilde{p}$ is small, then Lemma 1 can be used to reduce the error to at most $1/\tilde{p}^t$ for a suitable value of $t$. However, this comes at the cost of a multiplicative factor $t$ in efficiency. We now describe another construction that achieves an error rate of $1/\tilde{p}^t$ that comes at the cost of just an *additive* factor of $O(t)$ in efficiency.

Let $t \geq 1$ be fixed, and let $E$ be an arbitrary finite set. Our construction yields a projective hash family $\hat{\mathbf{H}}$ for $(X \times E, L \times E)$. We use the same name $\hat{\mathbf{H}}$ for this projective hash family as in the construction of Theorem 3, because when $t = 1$, the constructions are identical. Fix an injective encoding function

$$\Gamma : X \times E \to \{0, \ldots, \tilde{p} - 1\}^n,$$

where $n$ is sufficiently large.

Let $\hat{\mathbf{H}} = (\hat{H}, K^{n+2t-1}, X \times E, L \times E, \Pi, S^{n+2t-1}, \hat{\alpha})$, where $\hat{H}$ and $\hat{\alpha}$ are defined as follows. For

$$\vec{k} = (k'_1, \ldots, k'_t,\ k_1, \ldots, k_{n+t-1}) \in K^{n+2t-1},$$

$x \in X$, and $e \in E$, we define

$$\hat{H}_{\vec{k}}(x, e) = (\pi_1, \ldots, \pi_t),$$

where

$$\pi_j = H_{k'_j}(x) + \sum_{i=1}^{n} \gamma_i H_{k_{i+j-1}}(x) \qquad (j = 1, \ldots, t),$$

and $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e)$. We also define

$$\hat{\alpha}(\vec{k}) = (\alpha(k'_1), \ldots, \alpha(k'_t),\ \alpha(k_1), \ldots, \alpha(k_{n+t-1})).$$

Again, it is clear that $\hat{\mathbf{H}}$ is a projective hash family.

**Theorem 4** *Let $\hat{\mathbf{H}}$ be as above. Let $\vec{s} \in \alpha(K)^{n+2t-1}$, $x, x^* \in X$, and $e, e^* \in E$ be fixed, where $(x, e) \neq (x^*, e^*)$. Consider the probability space defined by choosing $\vec{k} \in \hat{\alpha}^{-1}(\vec{s})$ at random, and let $\vec{\pi} = \hat{H}_{\vec{k}}(x, e)$ and $\vec{\pi}^* = \hat{H}_{\vec{k}}(x^*, e^*)$. Then $\vec{\pi}$ is uniformly distributed over a coset of $\mathcal{I}(x)^t$ in $\Pi^t$ (the precise coset depending on $s$, $x$, and $e$), and $\vec{\pi}^*$ is uniformly and independently distributed over a coset of $\mathcal{I}(x^*)^t$ in $\Pi^t$ (the precise coset depending on $s$, $x^*$, and $e^*$). In particular, if the underlying group system $\mathbf{G}$ is diverse, then $\hat{\mathbf{H}}$ is $1/\tilde{p}^t$-universal$_2$.*

Proof. Let $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e)$, and $(\gamma_1^*, \ldots, \gamma_n^*) = \Gamma(x^*, e^*)$. Let

$$\vec{\rho} = (H_{k'_1}, \ldots, H_{k'_t},\ H_{k_1}, \ldots, H_{k_{n+t-1}}) \in \mathcal{H}^{n+2t-1}.$$

Now define the matrix $M \in \mathbb{Z}^{2t \times (n+2t-1)}$ as

$$M = \underbrace{\left(\begin{array}{cccc}1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \\ 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1\end{array}\right.}_{t \text{ columns}}\underbrace{\left.\begin{array}{cccccccc}\gamma_1 & \gamma_2 & & \cdots & \gamma_n & & & \\ & \gamma_1 & \gamma_2 & & \cdots & \gamma_n & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \gamma_1 & \gamma_2 & & \cdots & \gamma_n \\ \gamma_1^* & \gamma_2^* & & \cdots & \gamma_n^* & & & \\ & \gamma_1^* & \gamma_2^* & & \cdots & \gamma_n^* & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \gamma_1^* & \gamma_2^* & & \cdots & \gamma_n^*\end{array}\right)}_{n+t-1 \text{ columns}}$$

so that if

$$(\tilde{\rho}_1, \ldots, \tilde{\rho}_t,\ \tilde{\rho}_1^*, \ldots, \tilde{\rho}_t^*)^\top = M\vec{\rho}^\top,$$

then

$$\vec{\pi} = (\tilde{\rho}_1(x), \ldots, \tilde{\rho}_t(x)) \ \text{ and } \ \vec{\pi}^* = (\tilde{\rho}_1^*(x), \ldots, \tilde{\rho}_t^*(x)).$$

*Claim.* The rows of $M$ are linearly independent modulo $p$ for any prime $p$ dividing $|\mathcal{A}(L)|$.

The theorem is implied by the claim, as we now argue. By Lemma 9, the map $\mathbf{T}(M, \mathcal{A}(L))$ is surjective. By Lemma 8, $\vec{\rho}$ is uniformly distributed over a coset of $\mathcal{A}(L)^{n+2t-1}$ in $\mathcal{H}^{n+2t-1}$. It follows that $(\tilde{\rho}_1, \ldots, \tilde{\rho}_t,\ \tilde{\rho}_1^*, \ldots, \tilde{\rho}_t^*)$ is uniformly distributed over a coset of $\mathcal{A}(L)^{2t}$ in $\mathcal{H}^{2t}$, and therefore, $\vec{\pi}$ and $\vec{\pi}^*$ are uniformly and independently distributed over cosets of $\mathcal{I}(x)^t$ and $\mathcal{I}(x^*)^t$, respectively, in $\Pi^t$.

That proves the first statement of the theorem. The second statement of the theorem now follows from Lemma 7.

So now it remains to prove the above claim. Fix a prime $p$ dividing $|\mathcal{A}(L)|$, and for $1 \leq i \leq n$, let $\bar{\gamma}_i$ and $\bar{\gamma}_i^*$ denote the images of $\gamma_i$ and $\gamma_i^*$, respectively, in $\mathbb{Z}_p$, and let $\bar{M}$ denote the image of $M$ in $\mathbb{Z}_p^{2t \times (n+2t-1)}$. By the definition of $\Gamma$ and Lemma 6, we know that $\bar{\gamma}_i \neq \bar{\gamma}_i^*$ for some $1 \leq i \leq n$; let $i'$ be the least such $i$.

Now, suppose that

$$(c_1, \ldots, c_t,\ d_1, \ldots, d_{n+t-1}) = (a_1, \ldots, a_t,\ b_1, \ldots, b_t)\bar{M},$$

for

$$c_1, \ldots, c_t,\ d_1, \ldots, d_{n+t-1},\ a_1, \ldots, a_t,\ b_1, \ldots, b_t \ \in \mathbb{Z}_p.$$

Further suppose that $c_1, \ldots, c_t,\ d_1, \ldots, d_{n+t-1}$ are all zero. To prove the claim, we need to show that $a_1, \ldots, a_t,\ b_1, \ldots, b_t$ are all zero as well. It is clear from the structure of the matrix $M$, and since $c_1, \ldots, c_t$ are all zero, that we must have $a_j = -b_j$ for all $1 \leq j \leq t$. By way of contradiction, suppose that some $a_j \neq 0$ for some $1 \leq j \leq t$, and let $j'$ be the least such $j$. By direct calculation, one sees that

$$d_{i'+j'-1} = a_{j'}(\bar{\gamma}_{i'} - \bar{\gamma}_{i'}^*) \neq 0,$$

which is a contradiction. That proves the claim. $\triangle$

## 7.4 Examples of diverse group systems

In this section, we discuss two examples of diverse group systems that have cryptographic importance.

### 7.4.1 Example 1

Let $G$ be a group of prime of prime order $q$, and let $X = G^r$, i.e., $X$ is the direct product of $r$ copies of $G$. Let $L$ be any proper subgroup of $X$, and let $\mathcal{H} = \mathrm{Hom}(X, G)$. Consider the group system $\mathbf{G} = (\mathcal{H}, X, L, G)$.

The group $X$ is isomorphic as a $\mathbb{Z}_q$-vector space to $\mathbb{Z}_q^r$. For the purposes of this discussion, let us simply identify $X$ with $\mathbb{Z}_q^r$ and $G$ with $\mathbb{Z}_q$. Under this identification, $L$ is a proper $\mathbb{Z}_q$-subspace of $X$. Moreover, $\mathcal{H}$ can be identified with the vector space $\mathbb{Z}_q^r$, as follows: for every $\nu \in \mathbb{Z}_q^r$, we define $\phi_\nu \in \mathcal{H}$ to be the map that sends $x \in \mathbb{Z}_q^r$ to $(x, \nu) \in \mathbb{Z}_q$, where $(\cdot, \cdot)$ denotes the standard inner product of vectors.

For any set $U \subset \mathbb{Z}_q^r$, $\mathcal{A}(U)$ is the orthogonal complement in $\mathbb{Z}_q^r$ of the subspace of $\mathbb{Z}_q^r$ generated by $U$. Therefore, if $U$ generates a subspace of dimension $a$, $\mathcal{A}(U)$ is a subspace dimension $r - a$.

Now suppose $L$ has dimension $d$, and that $x \in X \setminus L$. It follows $\mathcal{A}(L)$ has dimension $r - d$, and $\mathcal{A}(L \cup \{x\})$ has dimension $r - d - 1$. This shows that $\mathbf{G}$ is diverse. Moreover, for any $x \in X \setminus L$, we have $\mathcal{I}(x) = \mathcal{E}_x(\mathcal{A}(L)) = \mathbb{Z}_q$. Therefore, a projective hash family derived from $\mathbf{G}$ is $1/q$-universal, or equivalently, 0-smooth.

#### 7.4.2 Example 2

Let $X$ be a cyclic group of order $a = bb'$, where $b' > 1$ and $\gcd(b, b') = 1$, and let $L$ be the unique subgroup of $X$ of order $b$. Let $\mathcal{H} = \mathrm{Hom}(X, X)$, and consider the group system $\mathbf{G} = (\mathcal{H}, X, L, X)$.

The group $X$ is isomorphic to $\mathbb{Z}_a$. If we identify $X$ with $\mathbb{Z}_a$, then $\mathcal{H}$ can be identified with $\mathbb{Z}_a$ as follows: for every $\nu \in \mathbb{Z}_a$, define $\phi_\nu \in \mathcal{H}$ to be the map that sends $x \in \mathbb{Z}_a$ to $x \cdot \nu \in \mathbb{Z}_a$.

The group $X$ is of course also isomorphic to $\mathbb{Z}_b \times \mathbb{Z}_{b'}$. If we identify $X$ with $\mathbb{Z}_b \times \mathbb{Z}_{b'}$, then $L$ corresponds to $\mathbb{Z}_b \times \langle 0 \rangle$. Moreover, we can identify $\mathcal{H}$ with $\mathbb{Z}_b \times \mathbb{Z}_{b'}$ as follows: for $(\nu, \nu') \in \mathbb{Z}_b \times \mathbb{Z}_{b'}$, let $\psi_{\nu,\nu'} \in \mathcal{H}$ be the map that sends $(x, x') \in \mathbb{Z}_b \times \mathbb{Z}_{b'}$ to $(x \cdot \nu, x' \cdot \nu') \in \mathbb{Z}_b \times \mathbb{Z}_{b'}$.

Under the identification in the previous paragraph, it is evident that $\mathcal{A}(L)$ is the subgroup of $\mathcal{H}$ generated by $\psi_{0,1}$. If we take any $(x, x') \in X \setminus L$, so that $x' \neq 0$, we see that $\psi_{0,1}(x, x') = (0, x')$. Thus, $\psi_{0,1} \notin \mathcal{A}(L \cup \{(x, x')\})$, which shows that $\mathbf{G}$ is diverse. Therefore, a projective hash family derived from $\mathbf{G}$ is $1/\tilde{p}$-universal, where $\tilde{p}$ is the smallest prime dividing $b'$.

It is also useful to characterize the group $\mathcal{I}(x, x') = \mathcal{E}_{x,x'}(\mathcal{A}(L))$. Evidently, since $\mathcal{A}(L) = \langle \psi_{0,1} \rangle$, we must have $\mathcal{I}(x, x') = \langle 0 \rangle \times \langle x' \rangle$.

# 8 Concrete encryption schemes

We present two new public-key encryption schemes secure against adaptive chosen ciphertext attack. These are derived from the general construction in §6, although we also present several variations that do not quite fit into this framework.

The first scheme is based on Paillier's Decision Composite Residuosity assumption. Ours is the first practical public-key encryption scheme secure against adaptive chosen ciphertext attack under this assumption.

The second is based on the classical Quadratic Residuosity assumption. Ours is the first public-key encryption scheme secure against adaptive chosen ciphertext attack under this assumption that is at all practical, as opposed to theoretical constructions such as [DDN].

Before presenting the new schemes, we show how the public-key encryption scheme from [CS] can be viewed as a special case of our general construction.

## 8.1 Schemes based on the Decision Diffie-Hellman assumption

### 8.1.1 Derivation

We show how to derive a secure encryption scheme based on the Decision Diffie-Hellman assumption from our generic encryption scheme construction in §6, together with our general techniques for building universal projective hash families in §7.

*The DDH assumption.* Let $G$ be a group of given large prime order $q$. We shall use additive notation for $G$, and view $G$ as a $\mathbb{Z}_q$-module in the natural way. The Decision Diffie-Hellman

(DDH) assumption is the assumption that it is hard to distinguish tuples of the form

$$(g_0, g_1, rg_0, r'g_1)$$

from tuples of the form

$$(g_0, g_1, rg_0, rg_1),$$

where $g_1$ and $g_1$ are randomly chosen from $G$, and $r$ and $r'$ are randomly chosen from $\mathbb{Z}_q$.

To be completely formal, one should actually specify a sequence of distributions of groups, such that for each value of a security parameter $\ell \geq 0$, a description of a group $G$, together with $q$, can be efficiently sampled from some distribution parameterized by $\ell$. Also, for such a group $G$, each element of the group should have a unique, compact binary encoding, and it should be the case that valid binary encodings of group elements are easily recognizable, that the group operation can be efficiently implemented, and that random elements of $G$ can be efficiently generated. We assume that $1/q$ is bounded by $\epsilon = \epsilon(\ell)$ for all groups associated with security parameter $\ell$, where $\epsilon(\ell)$ is a negligible function in $k$.

There are many possible realizations of suitable groups $G$. For instance, let $p$ be a large prime, and let $q$ be a large prime factor of $p - 1$. Then $G$ is the unique sub-group of order $q$ in $\mathbb{Z}_p^*$. Alternatively, we can choose $G$ as a prime-order subgroup of the group defined by an elliptic curve.

*A subset membership problem.* With $G$ and $q$ given, we now define an instance of a subset membership problem as follows. Let $g_0$ and $g_1$ be randomly chosen elements of $G$. Define $X = G \times G$, and let $L$ be the subgroup of $X$ generated by $(g_0, g_1) \in X$. A *witness* for $(x_0, x_1) \in L$ is $w \in \mathbb{Z}_q$ such that $(x_0, x_1) = (wg_0, wg_1)$. The instance description $\Lambda$ consists of descriptions of $G$, $q$, $g_0$, and $g_1$.

Obviously, one can efficiently sample a random element of $L$, together with a corresponding witness, by generating $w \in \mathbb{Z}_q$ at random, and computing $(x_0, x_1) = (wg_0, wg_1)$.

It is clear that this defines a subset membership problem, and that the hardness of this subset membership problem is implied by the DDH assumption for $G$.

*Hash proof systems.* Now it remains to construct appropriate strongly smooth and strongly universal$_2$ HPS's for the construction in §6. To do this, we first construct a diverse group system (see Definition 10), from which we can then derive the required HPS's.

Fix an instance description $\Lambda$, where $\Lambda$ specifies a group $G$ of order $q$, along with $g_0, g_1 \in G$, and let $X$ and $L$ be groups as defined above. Let $\mathcal{H} = \mathrm{Hom}(X, G)$, and consider the group system $\mathbf{G} = (\mathcal{H}, X, L, G)$. As shown in §7.4.1, $\mathbf{G}$ is a diverse group system.

Let $K = \mathbb{Z}_q \times \mathbb{Z}_q$, and for $(k_0, k_1) \in K$, let $H_{k_0,k_1} \in \mathrm{Hom}(X, G)$ be the map that sends $(x_0, x_1) \in X$ to $k_0x_0 + k_1x_1 \in G$. As discussed in §7.4.1, the correspondence $(k_0, k_1) \mapsto H_{k_0,k_1}$ is a bijection between $K$ and $\mathrm{Hom}(X, G)$.

Consider the projective projective hash family $\mathbf{H} = (H, K, X, L, G, G, \alpha)$, where $H$ and $K$ are as in the previous paragraph, and $\alpha$ maps $(k_0, k_1) \in K$ to $H_{k_0,k_1}(g_0, g_1) = k_0g_0 + k_1g_1 \in G$. It is clear that $\mathbf{H}$ is a projective hash family derived from $\mathbf{G}$, and so by Theorem 2 is $1/q$-universal, or equivalently, 0-smooth.

This immediately yields a strongly smooth HPS $\mathbf{P}$ corresponding to $\mathbf{H}$ — one simply needs to verify that all the algorithms that must be provided by an HPS are available. This is rather straightforward, and we leave the details to the reader (see the remark in the paragraph following Definition 9).

So now we have a strongly smooth HPS $\mathbf{P}$ as needed for the construction in §6.

Applying the construction in Theorem 3 to $\mathbf{H}$, we obtain a $1/q$-universal$_2$ projective hash family $\hat{\mathbf{H}}$ for $(X \times G, L \times G)$, and from this, a corresponding strongly universal$_2$ HPS $\hat{\mathbf{P}}$. Again, it is straightforward to verify that all the necessary algorithms required by an HPS are available.

### 8.1.2 The encryption scheme

We now present in detail the encryption algorithm obtained from the HPS's $\mathbf{H}$ and $\hat{\mathbf{H}}$ above.

We describe the scheme in terms of a fixed group of $G$ of order $q$. The message space for the scheme is the group $G$.

Let $\Gamma : G \times G \times G \to \mathbb{Z}_q^n$ be an efficiently computable injective map for an appropriate $n \geq 1$.

**Key Generation**

Generate $g_0, g_1 \in G$ at random and choose

$$k_0, k_1,\ \tilde{k}_0, \tilde{k}_1,\ \hat{k}_{1,1}, \hat{k}_{1,1}, \ldots, \hat{k}_{n,0}, \hat{k}_{n,1}\ \in \mathbb{Z}_q$$

at random.

Compute

$$s = k_0\, g_0 + k_1\, g_1 \in G, \quad \tilde{s} = \tilde{k}_0\, g_0 + \tilde{k}_1\, g_1 \in G, \quad \hat{s}_i = \hat{k}_{i,0}\, g_0 + \hat{k}_{i,1}\, g_1 \in G \quad (i = 1, \ldots, n).$$

The public key is $(g_0, g_1;\ s;\ \tilde{s};\ \hat{s}_1, \ldots, \hat{s}_n)$.

The private key is $(k_0, k_1;\ \tilde{k}_0, \tilde{k}_1;\ \hat{k}_{1,1}, \hat{k}_{1,1}, \ldots, \hat{k}_{n,0}, \hat{k}_{n,1})$.

**Encryption**

To encrypt a message $m \in G$ under a public key as above, one does the following.

Choose $w \in \mathbb{Z}_q$ at random, and compute

$$x_0 = wg_0 \in G, \quad x_1 = wg_1 \in G, \quad \pi = ws \in G, \quad e = m + \pi \in G.$$

Compute

$$\hat{\pi} = w\tilde{s} + \sum_{i=1}^{n} w\gamma_i \hat{s}_i \in G,$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x_0, x_1, e) \in \mathbb{Z}_q^n$.

The ciphertext is $(x_0, x_1, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x_0, x_1, e, \hat{\pi}) \in G^4$ under a secret key as above, one does the following.

Compute

$$\hat{\pi}' = (\tilde{k}_0 + \sum_{i=1}^{n} \gamma_i \hat{k}_{i,0})x_0 + (\tilde{k}_1 + \sum_{i=1}^{n} \gamma_i \hat{k}_{i,1})x_1 \ \in G,$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x_0, x_1, e) \in \mathbb{Z}_q^n$.

Check whether $\hat{\pi}' = \hat{\pi}$; if not, then output reject and halt.

Compute

$$\pi = k_0 x_0 + k_1 x_1 \in G, \quad m = e - \pi \in G,$$

and output $m$.

Note that in the decryption algorithm, we are assuming that $x_0, x_1, e, \hat{\pi}$ are elements of $G$. This implicitly means that the decryption algorithm should test that this is the case, and otherwise reject the ciphertext. These tests may have a non-trivial computational cost, and so it is worth noting that the test that $\hat{\pi} \in G$ can be omitted, without changing the functionality of the decryption algorithm.

This is *precisely* the scheme that our general construction in §6 yields, although we have simplified a few expressions using trivial algebraic identities. Thus, the scheme is secure against adaptive chosen ciphertext attack, provided the DDH assumption holds. This scheme is essentially the encryption scheme presented in §5.3 of [CS], with just a few very minor differences.

*Minor variations.* To obtain a more efficient scheme, one could drop the requirement that $\Gamma$ is injective. This would allow us to use a smaller value of $n$, possibly $n = 1$, thereby obtaining a much more compact and efficient scheme. It is straightforward to adapt our general framework to show that if $\Gamma$ is a collision resistant hash function (CRHF), then we still get a scheme that is secure against adaptive chosen ciphertext attack.

With a somewhat more refined analysis, one can show that a universal one-way hash function (UOWHF) [NY1] suffices. This analysis requires some additional, special properties of the subset membership problem; namely, that elements of $X \setminus L$ can be efficiently sampled at random, and that given appropriate "trapdoor" information (in this case, the discrete logarithm of $g_1$ with respect to $g_0$), elements of $X \setminus L$ can be efficiently distinguished from elements of $L$. When $n = 1$, the resulting encryption scheme is the main encryption scheme presented in [CS], with just a few very minor differences.

## 8.2 Schemes based on the Decision Composite Residuosity assumption

### 8.2.1 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 $\lambda$ bits in length. Let $N = pq$ and $N' = p'q'$. Consider the group $\mathbb{Z}_{N^2}^*$ and the subgroup $P$ of $\mathbb{Z}_{N^2}^*$ consisting of all $N$th powers of elements in $\mathbb{Z}_{N^2}^*$.

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

To be completely formal, one should specify specify a sequence of bit lengths $\lambda(\ell)$, parameterized by a security parameter $\ell \geq 0$, and to generate an instance of the problem for security parameter $\ell$, 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.

The primes $p'$ and $q'$ are called Sophie Germain primes by mathematicians, while $p$ and $q$ are called strong (or safe) primes by cryptographers. 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 $\lambda$-bit number is Sophie Germain prime is $\Omega(1/\lambda^2)$. We shall assume 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. As will become evident, we need to restrict ourselves to strong primes for technical reasons. However, 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.

*A subset membership problem.* We can decompose $\mathbb{Z}^*_{N^2}$ as an internal direct product

$$\mathbb{Z}^*_{N^2} = G_N \cdot G_{N'} \cdot G_2 \cdot T,$$

where each group $G_\tau$ is a cyclic group of order $\tau$, and $T$ is the subgroup of $\mathbb{Z}^*_{N^2}$ generated by $(-1 \bmod N^2)$. This decomposition is unique, except for the choice of $G_2$ (there are two possible choices). For any $x \in \mathbb{Z}^*_{N^2}$, we can express $x$ uniquely as $x = x(G_N)x(G_{N'})x(G_2)x(T)$, where for each $G_\tau$, $x(G_\tau) \in G_\tau$, and $x(T) \in T$. Note that the element $\xi = (1 + N \bmod N^2) \in \mathbb{Z}^*_{N^2}$ has order $N$, i.e., it generates $G_N$, and that $\xi^a = (1 + aN \bmod N^2)$ for $0 \leq a < N$.

Define the map

$$\begin{array}{rcl} \theta : \mathbb{Z}^*_{N^2} & \to & \{\pm 1\}, \\ (a \bmod N^2) & \mapsto & (a \mid N), \end{array}$$

where $(\cdot \mid \cdot)$ is the Jacobi symbol. It is clear that $\theta$ is a group homomorphism.

Let $X$ be the kernel of $\theta$. It is easy to see that $X = G_N G_{N'} T$, since $|\mathbb{Z}^*_{N^2}/X| = 2$ and $T \subset X$. In particular, $X$ is a cyclic group of order $2NN'$. Let $L$ be the subgroup of $N$th 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 $\Lambda$ will contain $N$, along with a random generator $g$ for $L$. It is easy to generate such a $g$: choose a random $\mu \in \mathbb{Z}^*_{N^2}$, 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. It is easy to see that it satisfies all the basic requirements specified in §4. The reason for using $(X, L)$ instead of $(\mathbb{Z}^*_{N^2}, P)$ is that $\mathbb{Z}^*_{N^2}$ 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 $\mathbb{Z}^*_{N^2}$ (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 $\mathbb{Z}^*_{N^2}$ 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.

*Hash proof systems.* Now it remains to construct appropriate strongly smooth and strongly universal$_2$ HPS's for the construction in §6. To do this, we first construct a diverse group system (see Definition 10), from which we can then derive the required HPS's.

Fix an instance description $\Lambda$, where $\Lambda$ 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 $\mathbf{G} = (\mathcal{H}, X, L, X)$. As discussed in §7.4.2, $\mathbf{G}$ is a diverse group system; moreover, for $x \in X$, we have $\mathcal{I}(x) = \langle x(G_N) \rangle$; thus, for $x \in X \setminus L$, $\mathcal{I}(x)$ has order $p$, $q$, or $N$, according to whether $x(G_N)$ has order $p$, $q$, or $N$.

For $k \in \mathbb{Z}$, let $H_k \in \mathrm{Hom}(X, X)$ be the $k$th power map; that is, $H_k$ sends $x \in X$ to $x^k \in X$. Let $K_* = \{0, \ldots, 2NN' - 1\}$. As discussed in §7.4.2, the correspondence $k \mapsto H_k$ 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 $\alpha$ maps $k \in \mathbb{Z}$ to $H_k(g) \in L$. Clearly, $\mathbf{H}_*$ is a projective hash family derived from $\mathbf{G}$, and so by Theorem 2, 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 $\Lambda$ 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 (see Definition 5). It is also easy to verify that all of the algorithms that $\mathbf{P}$ should provide are available.

So we now have a $2^{-\lambda(\ell)}$-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 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.

According to Theorem 2, 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 $\mathcal{I}(x)$ in $X$. As discussed above, $\mathcal{I}(x) = \langle x(G_N) \rangle$, and so is a subgroup of $G_N$. Moreover, for random $x \in X \setminus L$, we have $\mathcal{I}(x) \neq G_N$ with probability at most $2^{-\lambda+1}$.

Now define the map

$$\begin{array}{rcl} \chi : \mathbb{Z}_{N^2} & \rightarrow & \mathbb{Z}_N, \\ (a + bN \bmod N^2) & \mapsto & (b \bmod N) \quad (0 \leq a, b < N). \end{array}$$

This map does not preserve any algebraic structure; however, the restriction of $\chi$ to any coset of $G_N$ in $X$ is a one-to-one map from that coset onto $\mathbb{Z}_N$. To see this, let $x = (a + bN \bmod N^2) \in X$, where $0 \leq a, b < N$, and note that we must have $\gcd(a, N) = 1$; for $0 \leq c < N$, we have $x\xi^c = (a + (ac + b)N \bmod N)$, 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 $\mathbb{Z}_N$.

Let us define $\mathbf{H}_*^\times = (H^\times, K_*, X, L, \mathbb{Z}_N, L, \alpha)$, where for $k \in \mathbb{Z}$, $H_k^\times = \chi \circ H_k$. 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 $\chi$. 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 $2^{-\lambda(\ell)+1}$-smooth HPS $\mathbf{P}^\times$.

We can apply the construction in Theorem 3 to $\mathbf{H}_*$, obtaining a $2^{-\lambda}$-universal$_2$ projective hash family $\hat{\mathbf{H}}_*$ for $(X \times \mathbb{Z}_N, L \times \mathbb{Z}_N)$. From $\hat{\mathbf{H}}_*$ we get a corresponding approximation $\hat{\mathbf{H}}$ (using $K$ in place of $K_*$), and from this we get a corresponding $2^{-\lambda(\ell)}$-universal$_2$ extended HPS $\hat{\mathbf{P}}$.

We could build our encryption scheme directly using $\hat{\mathbf{P}}$; however, we get more compact ciphertexts if we modify $\hat{\mathbf{H}}_*$ by passing its hash outputs through $\chi$, just as we did in building $\mathbf{H}_*^\times$, obtaining the analogous projective hash family $\hat{\mathbf{H}}_*^\times$ for $(X \times \mathbb{Z}_N, L \times \mathbb{Z}_N)$. From Theorem 4, and the above discussion, it is clear that $\hat{\mathbf{H}}_*^\times$ is also $2^{-\lambda}$-universal$_2$. From $\hat{\mathbf{H}}_*^\times$ we get a corresponding approximation $\hat{\mathbf{H}}^\times$ (using $K$ in place of $K_*$), and from this we get a corresponding $2^{-\lambda(\ell)}$-universal$_2$ extended HPS $\hat{\mathbf{P}}^\times$.

### 8.2.2 The encryption scheme

We now present in detail the encryption scheme obtained from the HPS's $\mathbf{P}^\times$ and $\hat{\mathbf{P}}^\times$ 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 $\mathbb{Z}_N$.

Let $X$, $L$, $\theta$, and $\chi$ be as defined above. Also, let $W = \{0, \ldots, \lfloor N/2 \rfloor\}$ and $K = \{0, \ldots, \lfloor N^2/2 \rfloor\}$, as above. Let $R = \{0, \ldots, 2^\lambda - 1\}$, and let $\Gamma : \mathbb{Z}_{N^2} \times \mathbb{Z}_N \to R^n$ be an efficiently computable injective map for an appropriate $n \geq 1$. For sufficiently large $\lambda$, $n = 7$ suffices.

**Key Generation**

Choose $\mu \in \mathbb{Z}_{N^2}^*$ at random and set $g = -\mu^{2N} \in L$.

Choose

$$k, \; \tilde{k}, \; \hat{k}_1, \ldots, \hat{k}_n \in K$$

at random, and compute

$$s = g^k \in L, \quad \tilde{s} = g^{\tilde{k}} \in L, \quad \hat{s}_i = g^{\hat{k}_i} \in L \quad (i = 1, \ldots, n).$$

The public key is $(g; \; s; \; \tilde{s}; \; \hat{s}_1, \ldots, \hat{s}_n)$.

The private key is $(k; \; \tilde{k}; \; \hat{k}_1, \ldots, \hat{k}_n)$.

**Encryption**

To encrypt a message $m \in \mathbb{Z}_N$ under a public key as above, one does the following.

Choose $w \in W$ at random, and compute

$$x = g^w \in L, \quad y = s^w \in L, \quad \pi = \chi(y) \in \mathbb{Z}_N, \quad e = m + \pi \in \mathbb{Z}_N.$$

Compute

$$\hat{y} = \tilde{s}^w \prod_{i=1}^{n} \hat{s}_i^{\gamma_i w} \in L, \quad \hat{\pi} = \chi(\hat{y}) \in \mathbb{Z}_N,$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$.

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in X \times \mathbb{Z}_N \times \mathbb{Z}_N$ under a secret key as above, one does the following.

Compute

$$\hat{y} = x^{\tilde{k} + \sum_{i=1}^{n} \gamma_i \hat{k}_i} \in X, \quad \hat{\pi}' = \chi(\hat{y}) \in \mathbb{Z}_N,$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$.

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute

$$y = x^k \in X, \quad \pi = \chi(y) \in \mathbb{Z}_N, \quad m = e - \pi \in \mathbb{Z}_N,$$

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 $x \in \mathbb{Z}_{N^2}^*$ and that $\theta(x) = 1$, and reject the ciphertext if this does not hold.

This is *precisely* the scheme that our general construction in §6 yields. Thus, the scheme is secure against adaptive chosen ciphertext attack, provided the DCR assumption holds.

*Minor variations.* As in §8.1, if we replace $\Gamma$ by a CRHF we get an even more efficient scheme with a smaller value of $n$, possibly even $n = 1$. Moreover, as in §8.1, a UOWHF suffices, although this requires a more involved analysis.

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.

### 8.2.3 Variation 1

We now describe a variation on the above scheme. This variation is a bit simpler (but only marginally more efficient) than the scheme in §8.2.2. This scheme does not quite fit into our general framework, but can nevertheless be proven secure using the same basic ideas. This variation demonstrates that some aspects of the design of the scheme in §8.2.2 were carefully crafted so as to make that scheme fit into the general framework, but are not really necessary. We use this variation, along with the one in §8.2.3, as motivation for exploring some natural extensions to our general encryption framework.

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 $\mathbb{Z}_N$.

Let $L' = G_{N'}$, and let $\chi$ be as defined as in §8.2.1. Also, let $W' = \{0, \dots, \lfloor N/4 \rfloor\}$ and $K = \{0, \dots, \lfloor N^2/2 \rfloor\}$. Let $R = \{0, \dots, 2^\lambda - 1\}$, and let $\Gamma : \mathbb{Z}_{N^2} \times \mathbb{Z}_N \to R^n$ be an efficiently computable injective map for an appropriate $n \geq 1$.

**Key Generation**

Choose $\mu \in \mathbb{Z}_{N^2}^*$ at random and set $g = \mu^{2N} \in L'$.

Choose

$$k,\ \tilde{k},\ \hat{k}_1, \dots, \hat{k}_n \in K$$

at random, and compute

$$s = g^k \in L', \quad \tilde{s} = g^{\tilde{k}} \in L', \quad \hat{s}_i = g^{\hat{k}_i} \in L' \quad (i = 1, \dots, n).$$

The public key is $(g;\ s;\ \tilde{s};\ \hat{s}_1, \dots, \hat{s}_n)$.

The private key is $(k;\ \tilde{k};\ \hat{k}_1, \dots, \hat{k}_n)$.

**Encryption**

To encrypt a message $m \in \mathbb{Z}_N$ under a public key as above, one does the following.

Choose $w \in W'$ at random, and compute

$$x = g^w \in L', \quad y = s^w \in L', \quad \pi = \chi(y) \in \mathbb{Z}_N, \quad e = m + \pi \in \mathbb{Z}_N.$$

Compute

$$\hat{y} = \tilde{s}^w \prod_{i=1}^{n} \hat{s}_i^{\gamma_i w} \in L', \quad \hat{\pi} = \chi(\hat{y}) \in \mathbb{Z}_N,$$

where $(\gamma_1, \dots, \gamma_n) = \Gamma(x, e) \in R^n$.

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in \mathbb{Z}_{N^2}^* \times \mathbb{Z}_N \times \mathbb{Z}_N$ under a secret key as above, one does the following.

Compute
$$\hat{y} = x^{\tilde{k} + \sum_{i=1}^{n} \gamma_i \hat{k}_i} \in \mathbb{Z}_{N^2}^*, \quad \hat{\pi}' = \chi(\hat{y}) \in \mathbb{Z}_N,$$
where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$.

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute
$$y = x^k \in \mathbb{Z}_{N^2}^*, \quad \pi = \chi(y) \in \mathbb{Z}_N, \quad m = e - \pi \in \mathbb{Z}_N,$$
and output $m$.

Note that in the decryption algorithm, we are assuming that $x \in \mathbb{Z}_{N^2}^*$, which implicitly means that the decryption algorithm should check that this is the case, and reject the ciphertext if this does not hold.

The only differences between this variation and the scheme in §8.2.2 are that in this variation, (1) $g$ is computed as $\mu^{2N}$, rather than as $-\mu^{2N}$, (2) $w$ is chosen at random from $W'$, rather than from $W$, and (3) the decryption algorithm checks that $x \in \mathbb{Z}_{N^2}^*$, but does not additionally check that $\theta(x) = 1$.

*Security analysis.* Since this scheme does not fit into our general framework, we have to analyze its security. This scheme is secure against adaptive chosen ciphertext attack, under the DCR assumption. To prove this, we briefly sketch how our general framework can be extended so that this scheme fits into the framework.

Let us first consider a generalization of the notion of a smooth projective hash family. Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and let $X' \subset X$. We define two random variables, $U_{X'}(\mathbf{H})$ and $V_{X'}(\mathbf{H})$, as follows. Consider the probability space defined by choosing $k \in K$ at random, $x \in X' \setminus L$ at random, and $\pi' \in \Pi$ at random. We set $U_{X'}(\mathbf{H}) = (x, s, \pi')$ and $V_{X'}(\mathbf{H}) = (x, s, \pi)$, where $s = \alpha(k)$ and $\pi = H_k(x)$. For $\epsilon \geq 0$, we say that $\mathbf{H}$ is $\epsilon$-smooth over $X'$ if $U_{X'}(\mathbf{H})$ and $V_{X'}(\mathbf{H})$ are $\epsilon$-close.

Let us next consider the following generalization of a subset membership problem. In this generalization, an instance description specifies sets $X$, $L$, and $W$, and the relation $R$ just as for an ordinary subset membership problem, but in addition specifies a set $X' \subset X$. The instance sampling algorithm should behave just as for an ordinary subset membership problem; also, just as for an ordinary subset membership problem, it should be easy to recognize valid encodings of elements of $X$ (but not necessarily $X'$). However, the subset sampling algorithm is a bit different from that of an ordinary subset membership problem: the distribution of the output $x$ should be statistically close to the uniform distribution on $X' \cap L$ (rather than $L$). Also, the notion of hardness for a generalized subset membership problem is slightly different from that for an ordinary subset membership problem: hardness means that it it computationally hard to distinguish random elements of $X' \setminus L$ from random elements of $X' \cap L$ (rather than to distinguish random elements of $X \setminus L$ from random elements of $L$).

We also generalize the notion of a hash proof system, as follows. A hash proof system $\mathbf{P}$ for a generalized subset membership problem $\mathbf{M}$ associates with each instance $\Lambda[X, L, W, R, X']$ of $\mathbf{M}$ a projective hash family $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$, as well as a finite set $S'$ and an *auxiliary function* $\alpha' : K \to S'$. As for a hash proof system for an ordinary subset membership problem, there should be efficient algorithms to sample random elements of $K$, and to recognize valid encodings of elements of $\Pi$; also, the private evaluation algorithm should efficiently compute $H_k(x)$ given the instance description $\Lambda$ along with $k \in K$ and $x \in X$. However, we do not require an efficient algorithm to compute $\alpha(k)$ given the instance description $\Lambda$ along with $k \in K$; rather, we only require an efficient algorithm to compute $\alpha'(k)$ given the instance description $\Lambda$ along with $k \in K$; moreover, we require that for all $k \in K$, the value of $\alpha(k)$ determines the value of $\alpha'(k)$. Also, the public evaluation algorithm should efficiently compute $H_k(x)$ given the instance description $\Lambda$ along with $\alpha'(k) \in S'$, $x \in X' \cap L$, and a witness $w \in W$ for $x$. Note that although the private evaluation algorithm should work for all $x \in X$, including $x \in X \setminus X'$, the public evaluation algorithm need only work for $x \in X' \cap L$, and need not work for $x \in L \setminus X'$.

The definitions of $\epsilon(\ell)$-universal and $\epsilon(\ell)$-universal$_2$ hash proof systems for ordinary subset membership problems extend verbatim to hash proof systems for generalized subset membership problems. However, in defining an $\epsilon(\ell)$-smooth hash proof system for a generalized subset membership problem, the requirement is that the underlying projective hash family is $\epsilon(\ell)$-smooth over $X'$.

It is easy to adapt the generic encryption scheme presented in §6 to work with generalized hash proof systems. We sketch how this is done. Let $\mathbf{M}$ be a generalized subset membership problem specifying a sequence $(I_\ell)_{\ell \geq 0}$ of instance distributions. Let $\mathbf{P}$ be a strongly smooth HPS for $\mathbf{M}$, and for fixed $\ell \geq 0$ and $\Lambda[X, L, W, R, X'] \in [I_\ell]$, let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be the associated projective hash family, and let $\alpha' : K \to S'$ be the associated auxiliary function. As in §6, we assume that $\Pi$ is an abelian group. Let $\hat{\mathbf{P}}$ be a strongly universal$_2$ extended HPS for $\mathbf{M}$, and for fixed $\ell \geq 0$ and $\Lambda[X, L, W, R, X'] \in [I_\ell]$, let $\hat{\mathbf{H}} = (\hat{H}, \hat{K}, X \times \Pi, L \times \Pi, \hat{\Pi}, \hat{S}, \hat{\alpha})$ be the associated projective hash family, and let $\hat{\alpha}' : \hat{K} \to \hat{S}'$ be the associated auxiliary function. For the key generation algorithm, we choose $k \in K$ and $\hat{k} \in \hat{K}$ at random, and compute $s' = \alpha'(k)$ and $\hat{s}' = \hat{\alpha}'(k)$; the public key is $(s', \hat{s}')$, and the private key is $(k, \hat{k})$. The encryption algorithm is almost the same as in §6; the only difference is that $x$ is chosen at random from $X' \cap L$, and the computations of $H_k(x)$ and $\hat{H}_{\hat{k}}(x, e)$ using the public evaluation algorithms of $\mathbf{P}$ and $\hat{\mathbf{P}}$ make use of the values $s'$ and $\hat{s}'$. The decryption algorithm is identical to that in §6.

It is easy to adapt the proof of Theorem 1 to show that this scheme is secure against adaptive chosen ciphertext attack assuming the underlying generalized subset membership problem is hard. One uses the same simulator as in the proof of Theorem 1, except that now it is used to distinguish random elements of $X' \setminus L$ from random elements of $X' \cap L$. Except for this change, the proof of Theorem 1 carries through *verbatim*.

We now show how our variation of the DCR-based scheme fits into the above extended framework and is secure under the DCR assumption.

We first describe the generalized subset membership problem. For $N$ as above, let $X = \mathbb{Z}_{N^2}^*$, $L = G_{N'} G_2 T$, and let $X' = G_N G_{N'}$. Note that $X' \cap L = G_{N'} = L'$. An instance description $\Lambda$ will contain $N$, along with a generator $g$ for $L'$. To generate such a $g$, one can simply choose $\mu \in \mathbb{Z}_{N^2}^*$ at random, and compute $g = \mu^{2N}$. The set of witnesses is $W'$ as defined as above, and we say that $w \in W'$ is a witness for $x \in X$ if $x = g^w$. It is clear that if we choose $w \in W'$ at random, and set $x = g^w$, then we get a nearly random $x \in L'$ together with a corresponding witness $w \in W'$. That completes the description of our generalized subset membership problem.

The DCR assumption implies the hardness of this generalized subset membership problem. Indeed, recall that $P$ is the subgroup of $N$th powers of $\mathbb{Z}_{N^2}^*$. Evidently, $P = G_{N'} G_2 T = L$. Suppose we are given $x$ sampled at random from $\mathbb{Z}_{N^2}^*$ (respectively, $P$); then $x^2$ is uniformly distributed over $X'$ (respectively, $L'$). This implies that distinguishing $X'$ from $L'$ is at least as hard as distinguishing $\mathbb{Z}_{N^2}^*$ 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.

For $k \in \mathbb{Z}$, let $H_k$ be the $k$th power map on $X$; that is, $H_k$ maps $x \in X$ to $x^k \in X$. Consider the group system $\mathbf{G} = (\mathcal{H}, X, L, X)$, where $\mathcal{H} = \{H_k : k \in \mathbb{Z}\}$. Let $K_* = \{0, \ldots, 2NN' - 1\}$. Since $X$ has exponent $2NN'$, we see that the correspondence $k \mapsto H_k$ yields a bijection between $K_*$ and $\mathcal{H}$. We leave it to the reader to verify that $\mathbf{G}$ is diverse, and moreover, for any $x \in X$, $\mathcal{I}(x) = \langle x(G_N) \rangle$.

Consider the derived projective hash family $\mathbf{H}_* = (H, K_*, X, L, X, S, \alpha)$, where $H$ and $K_*$ are as in the previous paragraph, $S = L' \times G_2 \times T$, and for $k \in \mathbb{Z}$, we define $\alpha(k) = (g^k, g_1^k, g_2^k)$, where $g_1$ generates $G_2$ and $g_2 = (-1 \bmod N^2)$ generates $T$. In building a hash proof system from $\mathbf{H}_*$, we also define the auxiliary function $\alpha'$ that sends $k \in \mathbb{Z}$ to $g^k \in L'$, and as usual, we use the set $K$ in place of $K_*$. Using $H_*$ as the starting point, one sees that the variation presented in this section follows from precisely the same line of reasoning as in §8.2.1. It follows that the scheme is secure under the DCR assumption.

*Minor variations* As usual, instead of using an injective function $\Gamma$, we can use a CRHF, or even a UOWHF. In this case, we could typically take $n = 1$.

### 8.2.4 Variation 2

We describe another variation on the scheme in §8.2.2 that does not quite fit into our general framework, but can still be easily proven secure against adaptive chosen ciphertext attack using the techniques we have developed. In this variation, the ciphertexts are not as compact as those in the schemes in §8.2.2 and §8.2.3; however, the ciphertexts have more algebraic structure. A scheme such as this may be useful in certain applications.

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 $\mathbb{Z}_N$.

Let $X' = G_N G_{N'}$ and let $L' = G_{N'}$. Also, let $W' = \{0, \ldots, \lfloor N/4 \rfloor\}$ and $K = \{0, \ldots, \lfloor N^2/2 \rfloor\}$. Let $R = \{0, \ldots, 2^\lambda - 1\}$, and let $\Gamma : \mathbb{Z}_{N^2} \times \mathbb{Z}_{N^2} \to R^n$ be an efficiently computable injective map for an appropriate $n \geq 1$.

The key generation algorithm of this variation is identical to that of the scheme in §8.2.3. Only the encryption and decryption algorithms are different. Recall that $\xi = (1 + N \bmod N^2) \in \mathbb{Z}_{N^2}^*$ has order $N$, and that for $0 \leq a < N$, $\xi^a = (1 + aN \bmod N^2)$.

**Encryption**

To encrypt a message $m \in \mathbb{Z}_N$ under a public key as above, one does the following.

Choose $w \in W'$ at random, and compute

$$x = g^w \in L', \quad \pi = s^w \in L', \quad e = \xi^m \cdot \pi \in X'.$$

Compute

$$\hat{\pi} = \tilde{s}^w \prod_{i=1}^{n} \hat{s}_i^{\gamma_i w} \in L',$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$.

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in \mathbb{Z}_{N^2}^* \times \mathbb{Z}_{N^2}^* \times \mathbb{Z}_{N^2}^*$ under a secret key as above, one does the following.

Compute

$$\hat{\pi}' = x^{\tilde{k}+\sum_{i=1}^{n}\gamma_i \hat{k}_i} \in \mathbb{Z}_{N^2}^*,$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in R^n$.

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute

$$\pi = x^k \in X', \quad \tilde{m} = e \cdot \pi^{-1} \in \mathbb{Z}_{N^2}^*.$$

If $\tilde{m} = \xi^m$ for some $m \in \mathbb{Z}_N$, output $m$; otherwise, output reject.

Again, we implicitly assume that the decryption algorithm checks that $x$, $e$, and $\hat{\pi}$ lie in $\mathbb{Z}_{N^2}^*$. Clearly, however, the test that $x \in \mathbb{Z}_{N^2}^*$ (and $e \in \mathbb{Z}_{N^2}$ and $\hat{\pi} \in \mathbb{Z}_{N^2}$) is sufficient, since if $x \in \mathbb{Z}_{N^2}^*$, and either $e \notin \mathbb{Z}_{N^2}^*$ or $\hat{\pi} \notin \mathbb{Z}_{N^2}^*$, the ciphertext will anyway be rejected for other reasons.

*Security analysis.* This scheme is secure against adaptive chosen ciphertext attack, under the DCR assumption. To prove this, we briefly sketch how our general framework, as already extended in §8.2.3, can be further extended so that this scheme fits into the framework.

All we need to do is define an appropriate generalization of a smooth projective hash family. Let $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and let $X' \subset X$. Further, suppose that $\Pi$ is an abelian group (for which we use additive notation), and that $\Pi'$ is a subgroup of $\Pi$. We define two random variables, $U_{X'}^{\Pi'}(\mathbf{H})$ and $V_{X'}^{\Pi'}(\mathbf{H})$, as follows. Consider the probability space defined by choosing $k \in K$ at random, $x \in X' \setminus L$ at random, and $\pi' \in \Pi'$ at random. We set $U_{X'}^{\Pi'}(\mathbf{H}) = (x, s, \pi' + \pi)$ and $V_{X'}^{\Pi'}(\mathbf{H}) = (x, s, \pi)$, where $s = \alpha(k)$ and $\pi = H_k(x)$. For $\epsilon \geq 0$, we say that $\mathbf{H}$ is $\epsilon$-smooth over $X'$ on $\Pi'$ if $U_{X'}^{\Pi'}(\mathbf{H})$ and $V_{X'}^{\Pi'}(\mathbf{H})$ are $\epsilon$-close.

In building an encryption scheme using such a smooth projective hash family, $\Pi'$ will be the message space, rather than $\Pi$, and we require that it is easy to recognize valid binary encodings of elements of $\Pi'$.

We leave it to the reader to fill in all of the details of this extension, as well as to adapt the proof of Theorem 1 to this extension.

In the encryption scheme described above, we take $\Pi = \mathbb{Z}_{N^2}^*$ and $\Pi' = G_N$. We leave it to the reader to fill in the remaining details of the analysis of this scheme.

*Minor variations.* As usual, instead of using an injective function $\Gamma$, we can use a CRHF, or even a UOWHF. In this case, we could typically take $n = 1$.

## 8.3 Schemes based on the Quadratic Residuosity assumption

### 8.3.1 Derivation

*The QR 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 $\lambda$ bits in length. Let $N = pq$ and let $N' = p'q'$. Consider the group $\mathbb{Z}_N^*$, and let $X$ be the subgroup of elements $(a \bmod N) \in \mathbb{Z}_N^*$ with Jacobi symbol $(a \mid N) = 1$, and let $L$ be the subgroup of squares (a.k.a., quadratic residues) of $\mathbb{Z}_N^*$. Note that $L$ is a subgroup of $X$ of index 2.

The Quadratic Residuosity (QR) assumption is that given only $N$, it is hard to distinguish random elements of $X$ from random elements of $L$. This implies that it is hard to distinguish random elements of $X \setminus L$ from random elements of $L$.

To be completely formal, one should specify specify a sequence of bit lengths $\lambda(\ell)$, parameterized by a security parameter $\ell \geq 0$, and to generate an instance of the problem for security parameter $\ell$, 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.

As in §8.2, we shall assume that strong primes (such as $p$ and $q$) are sufficiently dense. Note that the traditional QR assumption was not restricted to strong primes. However, the QR assumption without this restriction implies the QR assumption with this restriction, assuming that strong primes are sufficiently dense, as we are here.

*A subset membership problem.* The groups $X$ and $L$ above will define our subset membership problem.

We can decompose $\mathbb{Z}_N^*$ as an internal direct product

$$\mathbb{Z}_N^* = G_{N'} \cdot G_2 \cdot T,$$

where each group $G_\tau$ is a cyclic group of order $\tau$, and $T$ is the subgroup of $\mathbb{Z}_N^*$ generated by $(-1 \bmod N)$. This decomposition is unique, except for the choice of $G_2$ (there are two possible choices).

It is easy to see that $X = G_{N'}T$, so it is a cyclic group, and that $L = G_{N'}$.

Our instance description $\Lambda$ will contain $N$, along with a random generator $g$ for $L$. It is easy to generate such a $g$: choose a random $\mu \in \mathbb{Z}_N^*$, and set $g = \mu^2$. 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 $W = \{0, \ldots, \lfloor N/4 \rfloor\}$. 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 is $O(2^{-\lambda})$-close to it.

This completes the description of our subset membership problem. It is easy to see that it satisfies all the basic requirements specified in §4. As already mentioned, the QR assumption implies that this is a hard subset membership problem.

*Hash proof systems.* Now it remains to construct appropriate strongly smooth and strongly universal$_2$ HPS's for the construction in §6. To do this, we first construct a diverse group system (see Definition 10), from which we can then derive the required HPS's.

Fix an instance description $\Lambda$, where $\Lambda$ 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 $\mathbf{G} = (\mathcal{H}, X, L, X)$.

As discussed in §7.4.2, $\mathbf{G}$ is a diverse group system; moreover, for $x \in X$, if we decompose $x$ as $x = x(L) \cdot x(T)$, where $x(L) \in L$ and $x(T) \in T$, then we have $\mathcal{I}(x) = \langle x(T) \rangle$; thus, for $x \in X \setminus L$, $\mathcal{I}(x) = T$.

For $k \in \mathbb{Z}$, let $H_k \in \mathrm{Hom}(X, X)$ be the $k$th power map; that is, $H_k$ sends $x \in X$ to $x^k \in X$. Let $K_* = \{0, \ldots, 2N' - 1\}$. As discussed in §7.4.2, the correspondence $k \mapsto H_k$ 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 $\alpha$ maps $k \in \mathbb{Z}$ to $H_k(g) \in L$. Clearly, $\mathbf{H}_*$ is a projective hash family derived from $\mathbf{G}$, and so by Theorem 2, it is 1/2-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 $\Lambda$ is slightly different than $\mathbf{H}_*$; namely, we use the set $K = \{0, \ldots, \lfloor N/2 \rfloor\}$ 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 1/2-universal HPS $\mathbf{P}$. We can apply the construction in Lemma 1 to $\mathbf{H}_*$, using a parameter $t = t(\ell)$, to get a $2^{-t}$-universal projective hash family $\bar{\mathbf{H}}_*$. From $\bar{\mathbf{H}}_*$ we get a corresponding approximation $\bar{\mathbf{H}}$ (using $K$ in place of $K_*$), and from this we get corresponding $2^{-t}$-universal HPS $\bar{\mathbf{P}}$.

Now, we could easily convert $\bar{\mathbf{P}}$ into a strongly smooth HPS by applying the Leftover Hash Lemma construction in Lemma 4 to the underlying projective hash family $\bar{\mathbf{H}}_*$. However, there is a much more direct and practical way to proceed, as we now describe.

According to Theorem 2, 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 $\mathcal{I}(x)$ in $X$. As discussed above, for $x \in X \setminus L$, $\mathcal{I}(x) = T$.

Now define the map $\chi : \mathbb{Z}_N \to \mathbb{Z}_2$ as follows: for $x = (a \bmod N) \in \mathbb{Z}_N^*$, with $0 \le a < N$, let $\chi(x) = 1$ if $a > N/2$, and $\chi(x) = 0$ otherwise. It is easy to verify that the restriction of $\chi$ to any coset of $T$ in $X$ (which is a set of the form $\{\pm x\}$ for some $x \in X$) is a one-to-one map from that coset onto $\mathbb{Z}_2$.

Let us define $\mathbf{H}_*^\times = (H^\times, K_*, X, L, \mathbb{Z}_N, L, \alpha)$, where for $k \in \mathbb{Z}$, $H_k^\times = \chi \circ H_k$. 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 $\chi$. From the observations in the previous two paragraphs, it is clear that $\mathbf{H}_*^\times$ is a 1/2-universal, and so 0-smooth, projective hash family.

Now, we can apply the construction in Lemma 1 to $\mathbf{H}_*^\times$ with the parameter $t = t(\ell)$ to get a 0-smooth projective hash family $\bar{\mathbf{H}}_*^\times$ whose hash output space is $\mathbb{Z}_2^t$. From $\bar{\mathbf{H}}_*^\times$ we get a corresponding approximation $\bar{\mathbf{H}}^\times$ (using $K$ in place of $K_*$), and from this we get corresponding 0-smooth HPS $\bar{\mathbf{P}}^\times$.

We can apply the construction in Theorem 4 to $\mathbf{H}_*$, using a parameter $\hat{t} = \hat{t}(\ell)$, obtaining a $2^{-\hat{t}}$-universal$_2$ projective hash family $\hat{\mathbf{H}}_*$ for $(X \times \mathbb{Z}_2^t, L \times \mathbb{Z}_2^t)$. From $\hat{\mathbf{H}}_*$ we get a corresponding approximation $\hat{\mathbf{H}}$ (using $K$ in place of $K_*$), and from this we get a corresponding $2^{-\hat{t}(\ell)}$-universal$_2$ extended HPS $\hat{\mathbf{P}}$.

We could build our encryption scheme directly using $\hat{\mathbf{P}}$; however, we get more compact ciphertexts if we modify $\hat{\mathbf{H}}_*$ by passing its hash outputs through $\chi$, just as we did in building $\mathbf{H}_*^\times$, obtaining the analogous projective hash family $\hat{\mathbf{H}}_*^\times$ for $(X \times \mathbb{Z}_2^t, L \times \mathbb{Z}_2^t)$. From Theorem 4, and the above discussion, it is clear that $\hat{\mathbf{H}}_*^\times$ is also $2^{-\hat{t}}$-universal$_2$. From $\hat{\mathbf{H}}_*^\times$ we get a corresponding approximation $\hat{\mathbf{H}}^\times$ (using $K$ in place of $K_*$), and from this we get a corresponding $2^{-\hat{t}(\ell)}$-universal$_2$ extended HPS $\hat{\mathbf{P}}^\times$.

### 8.3.2 The encryption scheme

We now present in detail the encryption obtained using the HPS's $\bar{\mathbf{P}}^\times$ and $\hat{\mathbf{P}}^\times$ above.

We describe the scheme for a fixed value of $N$ that is product of two $(\lambda+1)$-bit strong primes. The message space for this scheme is $\mathbb{Z}_2^t$, where $t = t(\ell)$ is an auxiliary parameter. Note that $t$ may be any size — it need not be particularly large. We also need an auxiliary parameter $\hat{t} = \hat{t}(\ell)$. The value of $\hat{t}$ should be large; more precisely, $2^{-\hat{t}(\ell)}$ should be a negligible function in $\ell$.

Let $X$, $L$, and $\chi$ be as defined above. Also as above, let $K = \{0, \ldots, \lfloor N/2 \rfloor\}$, and $W = \{0, \ldots, \lfloor N/4 \rfloor\}$. Let $\Gamma : \mathbb{Z}_N \times \mathbb{Z}_2^t \to \{0,1\}^n$ be an efficiently computable injective map for an appropriate $n \geq 1$.

**Key Generation**

Choose $\mu \in \mathbb{Z}_N^*$ at random and set $g = \mu^2 \in L$.

Randomly choose

$$k_1, \ldots, k_t, \quad \tilde{k}_1, \ldots, \tilde{k}_{\hat{t}}, \quad \hat{k}_1, \ldots, \hat{k}_{n+\hat{t}-1} \quad \in K.$$

Compute

$$\begin{aligned} s_i &= g^{k_i} \in L \quad (i = 1, \ldots, t), \\ \tilde{s}_i &= g^{\tilde{k}_i} \in L \quad (i = 1, \ldots, \hat{t}), \\ \hat{s}_i &= g^{\hat{k}_i} \in L \quad (i = 1, \ldots, n + \hat{t} - 1). \end{aligned}$$

The public key is $(g;\ s_1, \ldots, s_t;\ \tilde{s}_1, \ldots, \tilde{s}_{\hat{t}};\ \hat{s}_1, \ldots, \hat{s}_{n+\hat{t}-1})$.

The private key is $(k_1, \ldots, k_t;\ \tilde{k}_1, \ldots, \tilde{k}_{\hat{t}};\ \hat{k}_1, \ldots, \hat{k}_{n+\hat{t}-1})$.

**Encryption**

To encrypt a message $m \in \mathbb{Z}_2^t$ under a public key as above, one does the following.

Choose $w \in W$ at random, and compute

$$x = g^w, \quad y_i = s_i^w \in L \quad (i = 1, \ldots, t).$$

Compute

$$\pi = (\chi(y_1), \ldots, \chi(y_t)) \in \mathbb{Z}_2^t, \quad e = m + \pi \in \mathbb{Z}_2^t.$$

Compute

$$\begin{aligned} \tilde{z}_i &= \tilde{s}_i^w \in L && (i = 1, \ldots, t), \\ \hat{z}_i &= \hat{s}_i^w \in L && (i = 1, \ldots, n + \hat{t} - 1), \\ \hat{y}_i &= \tilde{z}_i \textstyle\prod_{j=1}^n (\hat{z}_{i+j-1})^{\gamma_j} \in L && (i = 1, \ldots, \hat{t}), \end{aligned}$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in \{0, 1\}^n$.

Compute

$$\hat{\pi} = (\chi(\hat{y}_1), \ldots, \chi(\hat{y}_{\hat{t}})) \in \mathbb{Z}_2^{\hat{t}}.$$

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in X \times \mathbb{Z}_2^t \times \mathbb{Z}_2^{\hat{t}}$ under a private key as above, one does the following.

Compute

$$\hat{y}_i = x^{\tilde{k}_i + \sum_{j=1}^n \gamma_j \hat{k}_{i+j-1}} \in X \quad (i = 1, \ldots, \hat{t}),$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in \{0, 1\}^n$.

Compute

$$\hat{\pi}' = (\chi(\hat{y}_1), \ldots, \chi(\hat{y}_{\hat{t}})) \in \mathbb{Z}_2^{\hat{t}}.$$

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute

$$y_i = x^{k_i} \in X \quad (i = 1, \ldots, t), \quad \pi = (\chi(y_1), \ldots, \chi(y_t)) \in \mathbb{Z}_2^t, \quad m = e - \pi \in \mathbb{Z}_2^t,$$

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 $x = (a \bmod N)$ with Jacobi symbol $(a \mid N) = 1$.

This is *precisely* the scheme that our general construction in §6 yields. Thus, the scheme is secure against adaptive chosen ciphertext attack, provided the QR assumption holds.

*Minor variations.* As in §8.1, if we replace $\Gamma$ by a CRHF we get an even more efficient scheme with a smaller value of $n$. In fact, just a UOWHF suffices.

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.

*Efficiency.* While this scheme is not nearly as efficient as our schemes based on the DDH and DCR assumptions, it is based on an assumption that is better established and qualitatively weaker than either of these assumptions. Moreover, the scheme may just be practical enough for some applications. Let us consider some concrete security parameters. We might choose $N$ to be a 1024-bit number. If we use this scheme just to encrypt a symmetric encryption key, then $t = 128$ is a reasonable value. Setting $\hat{t} = 128$ is also reasonable. If we implement $\Gamma$ using a hash function like SHA-1, then we can take $n = 160$.

With these choices of parameters, the size of a public or private key will be less than 70KB. Ciphertexts are quite compact, requiring 160 bytes. An encryption takes less than 600 1024-bit exponentiations modulo $N$; this will take about 10 seconds or so on typical a 1GHz PC. A decryption will require about half as many exponentiations modulo $N$, and so without any optimizations, this would take roughly half as much time as encryption; however, if we use the Chinese Remaindering optimizations mentioned above, this should cut the running time further by a factor of between 3 and 4; also, if we exploit the fact that all exponentiations in the decryption algorithm are to the same basis, further significant optimizations are possible, bringing the time for a decryption down to around one second or less.

So clearly, this scheme is not suitable for, say, implementation on a smart card. However, it is not astronomically impractical, either.

### 8.3.3 A variation

We now describe a variation on the above scheme. This variation is analogous to the variation of our basic DCR-based scheme, described in §8.2.4. The ciphertexts in this scheme are much less compact than those in the scheme above in §8.3.2, but have more algebraic structure, which may be useful in some applications.

We describe the scheme for a fixed value of $N$ that is product of two $(\lambda+1)$-bit strong primes. The message space for this scheme is $\mathbb{Z}_2^t$, where $t = t(\ell)$ is an auxiliary parameter. We also need an auxiliary parameter $\hat{t} = \hat{t}(\ell)$, where $2^{-\hat{t}(\ell)}$ is a negligible function in $\ell$.

Let $X$ and $L$ be as defined in §8.3.1. Also as in §8.3.1, let $K = \{0, \ldots, \lfloor N/2 \rfloor\}$, and $W = \{0, \ldots, \lfloor N/4 \rfloor\}$. Let $\Gamma : \mathbb{Z}_N \times \mathbb{Z}_N^t \to \{0,1\}^n$ be an efficiently computable injective map for an appropriate $n \geq 1$.

The key generation algorithm is the same as that in §8.3.2. We describe only the encryption and decryption algorithms.

**Encryption**

To encrypt a message $m = (m_1, \ldots, m_t) \in \mathbb{Z}_2^t$ under a public key as above, one does the following.

Choose $w \in W$ at random, and compute

$$x = g^w, \quad y_i = s_i^w \in L \quad (i = 1, \ldots, t).$$

Compute

$$e = ((-1)^{m_1} y_1, \ldots, (-1)^{m_t} y_t) \in X^t.$$

Compute

$$\begin{array}{rcll} \tilde{z}_i & = & \tilde{s}_i^w \in L & (i = 1, \ldots, t), \\ \hat{z}_i & = & \hat{s}_i^w \in L & (i = 1, \ldots, n + \hat{t} - 1), \\ \hat{y}_i & = & \tilde{z}_i \prod_{j=1}^n (\hat{z}_{i+j-1})^{\gamma_j} \in L & (i = 1, \ldots, \hat{t}), \end{array}$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in \{0, 1\}^n$.

Compute

$$\hat{\pi} = (\hat{y}_1, \ldots, \hat{y}_{\hat{t}}) \in L^{\hat{t}}.$$

The ciphertext is $(x, e, \hat{\pi})$.

**Decryption**

To decrypt a ciphertext $(x, e, \hat{\pi}) \in X \times X^t \times X^{\hat{t}}$ under a private key as above, one does the following.

Compute

$$\hat{y}_i = x^{\tilde{k}_i + \sum_{j=1}^n \gamma_j \hat{k}_{i+j-1}} \in X \quad (i = 1, \ldots, \hat{t}),$$

where $(\gamma_1, \ldots, \gamma_n) = \Gamma(x, e) \in \{0, 1\}^n$.

Compute

$$\hat{\pi}' = (\hat{y}_1, \ldots, \hat{y}_{\hat{t}}) \in X^{\hat{t}}.$$

Check whether $\hat{\pi} = \hat{\pi}'$; if not, then output reject and halt.

Compute

$$y_i = x^{k_i} \in X, \ \tilde{m}_i = y_i / e_i \in X \quad (i = 1, \ldots, t),$$

where $e = (e_1, \ldots, e_t)$. If for $1 \leq i \leq t$, $\tilde{m}_i$ is of the form $((-1)^{m_i} \bmod N)$ for some $m_i \in \mathbb{Z}_2$, then output $m = (m_1, \ldots, m_t)$; otherwise, output reject.

Note that in the decryption algorithm, we are assuming that $x \in X$, which implicitly means that the decryption algorithm should check that $x = (a \bmod N)$ with Jacobi symbol $(a \mid N) = 1$. It is sufficient to check that the components of $e$ and $\hat{\pi}$ are elements of $\mathbb{Z}_N$; if they are not elements of $X$ as well, the ciphertext will anyway be rejected.

It is easy to show that this scheme is secure under the QR assumption, using the extended framework sketched in §8.2.4 (one takes $\Pi = X$ and $\Pi' = T$ in the generalized smoothness definition), along with the analysis in §8.3.1. We leave the details to the reader.

As usual, instead of using an injective function $\Gamma$, we can use a CRHF, or even a UOWHF, allowing one to use a smaller value of $n$.

## Acknowledgments

Thanks to Ivan Damgaard for noting an improvement in the $1/p$-bound stated in Theorem 2, and thanks to Amit Sahai and Yehuda Lindell for useful discussions.

## References

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

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

[CW] J. Carter and M. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences,* 18:143–154, 1979.

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

[DDN] 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.

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

[NY1] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. STOC '89,* ACM Press, 1989.

[NY2] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proc. STOC '90,* ACM Press, 1990.

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

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

[WC] 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.