CCA2-Security Under Partial Key Exposure and the Cramer-Shoup Cryptosystem

Douglas Wikström

ETH Zürich, Department of Computer Science, douglas@inf.ethz.ch

Abstract. We introduce a stronger, but natural, variation of CCA2-security and observe that the generic CCA2-secure cryptosystem of Cramer and Shoup (1998) satisfy the stronger definition. We also present an alternative generic cryptosystem satisfying the stronger definition which is a combination of the schemes of Naor and Yung (1990), Cramer and Shoup (1998), and Sahai (1999). This is interesting, since it unifies these two seemingly independent approaches for constructing CCA2-secure cryptosystems. We also show how the stronger definition can be applied directly to significantly simplify, improve the efficiency, and reduce interaction in protocols such as mix-nets based on the El Gamal or Paillier cryptosystem.

1 Introduction

Several cryptographic protocols start with a submission phase where many parties submit ciphertexts encrypted with a public key pk, and a smaller group of servers holds shares of the secret key sk corresponding to pk. The servers then compute and publish some function of the input plaintexts. The problem with using a polynomially indistinguishable cryptosystem directly in such protocols is that it does not guarantee that the plaintexts submitted by corrupt parties are not related to those submitted by honest users.

Formally, the problem surfaces when the cryptographer constructing the protocol tries to reduce the security of his/her scheme to the security of the cryptosystem. If the simulation paradigm is used, some kind of simulator must be constructed and the simulator must extract the values of the corrupt parties to be able to hand these to an ideal version of the protocol, e.g., in the universal composability framework of Canetti [4]. The existence of a successful adversary must contradict the security of the cryptosystem, i.e., extraction must be done without using the secret key of the cryptosystem. This is not possible using a polynomially indistinguishable [12] cryptosystem naively.

1.1 Previous Work and Motivation

The solutions to the above problem found in the literature fall into two categories: the submitting party proves knowledge of the plaintext in zero-knowledge and a CCA2-secure cryptosystem is used. More precisely, the solutions can be characterized as follows:

1. The submitting party proves knowledge of its plaintext. There are three variations of this solution.

(a) An *interactive* proof of knowledge [13] is used, either with a straight-line extractor in the public key setting using the Naor and Yung [16] double-ciphertext trick, or with an extractor using rewinding.
(b) A *non-interactive* proof of knowledge is used. If based on a non-interactive zero-knowledge proof [2], this is not efficient for either the submittor or the receivers (even using the recent techniques of Groth et al. [14]). If based on the secret sharing based proof of knowledge exploiting the distributed public key setting [1], the computational and communication complexity of the sender grows linearly with the number of receivers.
(c) A *non-interactive* proof of knowledge in the *random oracle model* is used. This may be efficient, but it is only heuristically secure [5].

2. A CCA2-secure cryptosystem is used such that a ciphertext can be transformed into a new ciphertext for the polynomially indistinguishable scheme. Although in principle any CCA2-secure can be made distributed using general techniques the only concrete schemes we are aware of are the schemes of Canetti and Goldwasser [6] and Lysyanskaya and Peikert [15].
   These solutions are practical, but the communication complexity between the receivers is at least linear in the number of received messages.

To summarize, there is no solution to the above problem which has all the following nice properties at the same time: non-interactive submission, computational and communication complexity of submitter independent of number of secret key shares, communication complexity among receivers independent of number of the senders, and truly efficient.

Recall that to decrypt a Cramer-Shoup ciphertext it is first verified that it is valid. If it is valid, then decryption proceeds and otherwise the decryption algorithm simply outputs $\perp$. Furthermore, two distinct parts of the secret key are used to perform these two steps. Suppose for a moment that the scheme remains secure even if the first part of the key is disclosed to the adversary. Then the above problem could be solved by letting the servers reconstruct this part of the shared key, and simply check the validity of each ciphertext *without any interaction* with other servers, giving a significantly simplified and more efficient submission phase. This is the original motivation to this work.

1.2 Our Contribution

To capture the above scenario we introduce a stronger variant definition of CCA2-security, where part of the secret key is given to the adversary *before* it guesses the plaintext of the challenge ciphertext. Then we show that the generic scheme of Cramer and Shoup [9] satisfies this stronger definition. Then we describe an alternative generic scheme based on a combination of the constructions in Naor and Yung [16], Cramer and Shoup [9], and Sahai [20]. This unifies these two approaches to constructing CCA2-secure cryptosystems. Finally, we illustrate the usefulness of the new definition by applying it to mix-nets.

1.3 Notation

We denote by PT and PPT the sets of deterministic and probabilistic polynomial time Turing machines respectively, and write PT^* for the set of non-uniform polynomial time Turing machines. We use n to denote the security parameter, and say that a function $\epsilon(n)$ is negligible if for every constant c there exists a constant n_0 such that $\epsilon(n) < n^{-c}$ for $n > n_0$. If I is a distribution, then we denote by $[I]$ its support, i.e., the values assigned a positive probability. If pk is the public key of a cryptosystem, we denote by $\mathcal{M}_{pk}$ and $\mathcal{C}_{pk}$ the plaintext space and ciphertext space respectively.

2 The Idea

Recall the original construction of Cramer and Shoup [8]. The cryptosystem is deployed in a group G_q of prime order in which the decision Diffie-Hellman assumption is assumed to be hard. The key generator first chooses two random generators $g_0, g_1 \in G_q$. Then it choses random exponents $x_0, x_1, y_0, y_1, z \in \mathbb{Z}_q$ and defines $c = g_0^{x_0} g_1^{x_1}$, $d = g_0^{y_0} g_1^{y_1}$, and $h = g_0^z$. It also generates a collision-free hash function H. Finally, it outputs the pair of public and secret keys

$$(pk, sk) = ((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1, z)) \ .$$

To encrypt a message $m \in G_q$ using the public key pk the encryption algorithm chooses $r \in \mathbb{Z}_q$ randomly and outputs the tuple

$$(u_0, u_1, e, v) = (g_0^r, g_1^r, h^r m, c^r d^{rH(u_0,u_1,e)}) \ .$$

To decrypt a tuple $(u_0, u_1, e, v) \in G_q^4$ using the secret key sk the decryption algorithm first checks that $u_0^{x_0} u_1^{x_1} (u_0^{y_0} u_1^{y_1})^{H(u_0,u_1,e)} = v$. If so it outputs e/u_0^z, and otherwise $\perp$.

Note that $h = g^z$ and z have the form of an El Gamal [11] public and secret key respectively and that (u_0, e) is nothing more than an El Gamal ciphertext. This is of course not a new observation. What seems to be a new observation is the fact that the holder of the secret key may reveal (x_0, x_1, y_0, y_1) without any loss in security as long as it never decrypts any ciphertext constructed after this point, and that this is a very useful property.

3 CCA2-Security Under Partial Key Exposure

What today is known as CCA2-security was introduced by Rackoff and Simon [19], but the general idea of non-malleable and chosen ciphertext secure cryptosystems was developed by Naor and Yung [16], and Dolev, Dwork and Naor [10]. The breakthrough work in [10] gives the first construction of a CCA2-secure cryptosystem, but their construction is not particularly efficient. The first truly efficient CCA2-secure cryptosystem under standard complexity assumptions was presented by Cramer and Shoup [8].

Recall the game considered in the definition of CCA2-security. The adversary is given a public key pk. Then it may ask any number of decryption queries to a decryption oracle $\mathsf{Dec}_{sk}(\cdot)$ holding the secret key sk corresponding to pk. The adversary must then choose two challenge messages m_0 and m_1. The game flips a bit b and returns a challenge ciphertext on the form $c = \mathsf{Enc}_{pk}(m_b)$. Then the adversary is again allowed to ask arbitrary decryption queries to $\mathsf{Dec}_{sk}(\cdot)$ with the exception of c, and must finally output a guess b' of the bit b. If the cryptosystem is CCA2-secure, then the probability that that the guess is correct, i.e., $b' = b$, should be negligibly close to $1/2$ for every polynomial time adversary.

3.1 Partial Key Exposure

We want to capture the fact that part of the secret key may be given to the adversary provided that the decryption oracle does not answer any additional queries after this event. Thus, we need a way to denote which part of the key is published and which part stays secret. We simply assume that any secret key output by the key generator is a string on the form $sk = (sk_1 : sk_2)$.

The game we consider is identical to the original CCA2-security game except that the adversary is given the first part sk_1 of the secret key before it guesses the content of the challenge ciphertext, and after this point it is no longer allowed to ask any decryption queries. We call this CCA2-security under partial key exposure (CCA2-PKE-security). Clearly, any cryptosystem that is CCA2-secure can be trivially turned into one which is secure even with partial key exposures by simply putting the colon at the beginning of the secret key, but we are interested in non-trivial examples.

Experiment 1 (CCA2-Security Under PKE, $\mathsf{Exp}^{\mathsf{cca2-pke}-b}_{\mathcal{CS},A}(n)$).

$$
\begin{aligned}
(pk, (sk_1 : sk_2)) &\leftarrow \mathsf{CSKg}(1^n) \\
(m_0, m_1, \mathsf{state}_1) &\leftarrow A^{\mathsf{Dec}_{sk}(\cdot)}(\mathsf{choose}, pk) \\
c &\leftarrow \mathsf{Enc}_{pk}(m_b) \\
\mathsf{state}_2 &\leftarrow A^{\mathsf{Dec}_{sk}(\cdot)}(\mathsf{morequeries}, \mathsf{state}_1, c) \\
d &\leftarrow A(\mathsf{guess}, \mathsf{state}_2, sk_1)
\end{aligned}
$$

The experiment returns 0 if $\mathsf{Dec}_{sk}(\cdot)$ was queried on c, and d otherwise.

Definition 1 (CCA2-PKE-Security). *A public key cryptosystem $\mathcal{CS}$ is said to be secure against chosen ciphertext attacks under partial key exposure (CCA2-PKE-secure) if for all adversaries $A \in \mathrm{PT}^*$ the absolute value $|\Pr[\mathsf{Exp}^{\mathsf{cca2-pke}-0}_{\mathcal{CS},A}(n) = 1] - \Pr[\mathsf{Exp}^{\mathsf{cca2-pke}-1}_{\mathcal{CS},A}(n) = 1]|$ is negligible in n.*

Example 1. The most natural example is the original CCA2-secure cryptosystem described above. Simply redefine the key generation algorithm to output a pair on the form $((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1 : z))$.

4 Achieving CCA2-PKE-Security

The fact that the generic CCA2-secure cryptosystem of Cramer and Shoup remains CCA2-secure even under partial key exposure is quite easy to see from their security proof. On the other hand we need to show that this is indeed the case. Thus, we recall their scheme and prove this fact in the appendix, but we use coarse grained and streamlined definitions. We also take the liberty of ignoring the technical problem of constructing *efficiently computable* hash families, since this complicates the definitions and does not add anything to our exposition. See [9] for more details. Then we introduce an alternative generic construction and explain how the most efficient Cramer-Shoup schemes based on the El Gamal cryptosystem and the Paillier cryptosystem are instantiations of this.

4.1 Preliminaries

Subset Membership Problems. A subset membership problem consists of three sets X, $L \subsetneq X$, and W, and a relation $R \subset X \times W$. The idea is that it should be hard to decide if an element is sampled from $X \setminus L$ or from L. To be useful in cryptography we also need some algorithms that allow us to sample instances and elements, and check for membership in X. Formal definitions are given below.

Definition 2. *A subset membership problem* $\mathbf{M}$ *consists of a collection of distributions* $(I_n)_{n\in\mathbb{N}}$*, an instance generator* $\mathsf{Ins} \in \mathrm{PPT}$*, a sampling algorithm* $\mathsf{Sam} \in \mathrm{PPT}$*, and a membership checking algorithm* $\mathsf{Mem} \in \mathrm{PT}$ *such that:*

1. *I_n is a distribution on instance descriptions $\Lambda[X, L, W, R]$ specifying finite non-empty sets X, $L \subsetneq X$, and W, and a binary relation $R \subset X \times W$.*
2. *On input 1^n, Ins outputs an instance Λ with distribution statistically close to I_n.*
3. *On input 1^n and $\Lambda[X, L, W, R] \in [I_n]$, Sam outputs $(x, w) \in R$, where the distribution of x is statistically close to uniform over L.*
4. *On input 1^n, $\Lambda[X, L, W, R] \in [I_n]$, and $\zeta \in \{0,1\}^*$ Mem outputs 1 or 0 depending on if $\zeta \in X$ or not.*

Definition 3. *Let* $\mathbf{M}$ *be a subset membership problem. Sample Λ from I_n and let x and x' be randomly distributed over L and $X \setminus L$ respectively. We say that* $\mathbf{M}$ *is hard if for all $A \in \mathrm{PT}^*$ the absolute value $|\Pr[A(\Lambda, x) = 1] - \Pr[A(\Lambda, x') = 1]|$ is negligible.*

Hash Families. Hash families are well known in the cryptographic literature and there are many interesting variations. We assume implicitly that the families below are indexed by a security parameter n.

Definition 4. *A projective hash family* $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ *consists of finite non-empty sets K, X, $L \subsetneq X$, Π, and S, a function $\alpha : K \to S$, and a collection $H = (H_k : X \times \Pi \to \Pi)_{k\in K}$ of functions, where $\alpha(k)$ determines H_k on $L \times \Pi$.*

Definition 5. *Let* $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ *be a projective hash family, and let* $k \in K$ *be random. Then* $\mathbf{H}$ *is universal*$_2$ *if for all* $s \in S$, $x, x' \in X$, *and* $\pi_x, \pi'_x, \pi, \pi' \in \Pi$ *with* $x \notin L \cup \{x'\}$, $\Pr_k[H_k(x, \pi_x) = \pi \wedge H_k(x', \pi'_x) = \pi' \wedge \alpha(k) = s]$ *is negligible.*

The following lemma is stated implicitly in [9].

Lemma 1 (Projective Hashing). *Let* $\mathbf{H}$ *be a universal*$_2$ *projective hash family and consider the following experiment for an adversary* $A \in \mathrm{PT}^*$. *Let* τ_k *be the predicate defined by* $\tau_k((x, \pi_x), \pi) \iff H_k(x, \pi_x) = \pi$.

$$\begin{aligned} k &\leftarrow K \\ (x, \pi_x, \mathsf{state}) &\leftarrow A^{\tau_k(\cdot,\cdot)}(\alpha(k)) \\ &\leftarrow A^{\tau_k(\cdot,\cdot)}(H_k(x, \pi_x), \mathsf{state}) \end{aligned}$$

Denote by $((x_i, \pi_{x,i}), \pi_i)$ *the* i*th query to* τ_k, *and let* i_l *be the index of the last query before the first output. Denote by* E *the event that* A *asks a query* $((x_i, \pi_{x,i}), \pi_i)$ *to* τ_k *with* $H_k(x_i, \pi_{x,i}) = \pi_i$, $x_i \in X \setminus L$, *and* $i \leq i_l$ *or* $x_i \neq x$. *Then* $\Pr[E]$ *is negligible.*

Definition 6. *Let* $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$ *be a projective hash family, and let* $k \in K$, $x \in X \setminus L$, *and* $\pi \in X$ *be random. Then* $\mathbf{H}$ *is smooth if for every* $\pi_x \in \Pi$ *the distributions of* $(x, \pi_x, \alpha(k), H_k(x, \pi_x))$ *and* $(x, \pi_x, \alpha(k), \pi)$ *are statistically close.*

Universal Hash Proof Systems. Informally, a hash proof system may be viewed as a non-interactive zero-knowledge proof system where only the holder of a secret key corresponding to the public common random string can verify a proof. Strictly speaking, the definition below corresponds to a special case of what Cramer and Shoup [9] call "extended strongly (smooth, universal$_2$)" hash proof system.

Definition 7. *A (smooth, universal*$_2$*) hash proof system* $\mathbf{P}$ *for a subset membership problem* $\mathbf{M}$ *associates with each instance* $\Lambda[X, L, W, R]$ *a (smooth, universal*$_2$*) projective hash family* $\mathbf{H} = (H, K, X, L, \Pi, S, \alpha)$, *and the following algorithms*

1. *A key generation algorithm* $\mathsf{Gen} \in \mathrm{PPT}$ *that on input* 1^n *and* $\Lambda \in [I_n]$ *outputs* (s, k), *where* k *is randomly distributed in* K *and* $s = \alpha(k)$.
2. *A private evaluation algorithm* $\mathsf{PEval} \in \mathrm{PT}$ *that on input* 1^n, $\Lambda \in [I_n]$, $k \in K$, *and* $(x, \pi_x) \in X \times \Pi$ *outputs* $H_k(x, \pi_x)$.
3. *A public evaluation algorithm* $\mathsf{Eval} \in \mathrm{PT}$ *that on input* 1^n, $\Lambda \in [I_n]$, $\alpha(k)$ *with* $k \in K$, (x, π_x) *and* w, *with* $(x, w) \in R$, *outputs* $H_k(x, \pi_x)$.
4. *A membership checking algorithm* $\mathsf{Mem} \in \mathrm{PT}$ *that on input* 1^n, $\Lambda \in [I_n]$, *and* $\zeta \in \{0,1\}^*$ *outputs* 1 *or* 0 *depending on if* $\zeta \in \Pi$ *or not.*

4.2 The Generic Scheme of Cramer and Shoup

Given the definitions above it is not hard to describe the generic cryptosystem of Cramer and Shoup [9]. Let $\mathbf{M}$ be a hard subset membership problem, such that Π can be fitted with a group operation for any instance Λ, let $\mathbf{P}_0 = (\mathsf{Gen}_0, \mathsf{PEval}_0, \mathsf{Eval}_0, \mathsf{Mem}_0)$ be a smooth hash proof system for $\mathbf{M}$, and let $\mathbf{P}_1 = (\mathsf{Gen}_1, \mathsf{PEval}_1, \mathsf{Eval}_1, \mathsf{Mem}_1)$ be a universal$_2$ hash proof system for $\mathbf{M}$.

Key Generation. Compute $\Lambda[X, L, W, R] = \mathsf{Ins}(1^n)$, $(s, k) = \mathsf{Gen}_0(1^n, \Lambda)$, $(\widehat{s}, \widehat{k}) = \mathsf{Gen}_1(1^n, \Lambda)$, and output the key pair $(pk, sk) = ((\Lambda, \widehat{s}, s), (\widehat{k} : k))$.

Encryption of a message $m \in \Pi$**.** Compute $(x, w) = \mathsf{Sam}(\Lambda)$, $\pi = \mathsf{Eval}_0(\Lambda, s, x, w) = H_k(x)$, $e = m + \pi$, and $\widehat{\pi} = \mathsf{Eval}_1(\Lambda, \widehat{s}, x, w, e) = \widehat{H}_{\widehat{k}}(x, e)$ and output $(x, e, \widehat{\pi})$.

Decryption of a ciphertext $(x, e, \widehat{\pi})$**.** Output $m = e - \mathsf{PEval}_0(\Lambda, \widehat{k}, x) = e - H_k(x)$, only if $\mathsf{PEval}_1(\Lambda, \widehat{k}, x, e) = \widehat{H}_{\widehat{k}}(x, e) = \widehat{\pi}$ and otherwise output $\perp$.

We have not modified the cryptosystem in any way except that we have introduced a comma in the notation of the secret key to distinguish the two parts as needed in the CCA2-PKE-security definition. Cramer and Shoup prove (see Theorem 1 in [9]) that the above cryptosystem is CCA2-secure under the assumption that $\mathbf{M}$ is hard. We prove the slightly stronger result using their powerful machinery.

Proposition 1. *If* $\mathbf{M}$ *is a hard subset membership problem, then the generic Cramer-Shoup cryptosystem above is CCA2-PKE-secure.*

The only essential change to the proof is that one must observe that the projective hashing lemma is applicable even if $\widehat{k}$ is revealed at the end of the CCA2-experiment under partial key exposure, since no queries are asked after this point.

4.3 An Alternative Generic Scheme

There is a natural alternative way to construct a CCA2-secure cryptosystem which may be seen as a combination of the ideas of Naor and Yung [16], Cramer and Shoup [8], and Sahai [20]. This shows how these seemingly independent approaches are related.

Denote by $\mathcal{CS}^{(0)} = (\mathsf{CSKg}^{(0)}, \mathsf{Enc}^{(0)}, \mathsf{Dec}^{(0)})$ and $\mathcal{CS}^{(1)} = (\mathsf{CSKg}^{(1)}, \mathsf{Enc}^{(1)}, \mathsf{Dec}^{(1)})$ cryptosystems. Denote by $\mathbf{M} = ((I_n)_{n\in\mathbb{N}}, \mathsf{Ins}, \mathsf{Sam}, \mathsf{Mem})$ the subset membership problem defined as follows. To sample I_n, compute $(pk_0, sk_0) = \mathsf{CSKg}^{(0)}(1^n)$ and $(pk_1, sk_1) = \mathsf{CSKg}^{(1)}(1^n)$, and output (pk_0, pk_1). To define an instance $\Lambda[X, L, W, R]$ from these public keys, let $X = \mathcal{C}_{pk_0} \times \mathcal{C}_{pk_1}$, let $R = \{((c_0, c_1), (m, r_0, r_1)) \in \mathcal{C}_{pk_0} \times \mathcal{C}_{pk_1} \times \mathcal{M}_{pk_0} \cap \mathcal{M}_{pk_1} \times \{0,1\}^* \times \{0,1\}^* : c_0 = \mathsf{Enc}^{(0)}_{pk_0}(m, r_0) \wedge c_1 = \mathsf{Enc}^{(1)}_{pk_1}(m, r_1)\}$, and let L and W denote the projection of R onto its first and second component respectively. In other words X is the set of pairs of ciphertexts formed with pk_0 and pk_1, and the subspace L consists of all such pairs encrypting identical messages. Let $\mathbf{P} = (\mathsf{Gen}, \mathsf{PEval}, \mathsf{Eval}, \mathsf{Mem})$ be a smooth and universal$_2$ hash proof system for $\mathbf{M}$. We then define the CCA2-secure cryptosystem as follows.

Key Generation. Compute $(pk_0, sk_0) = \mathsf{CSKg}^{(0)}(1^n)$, $(pk_1, sk_1) = \mathsf{CSKg}^{(1)}(1^n)$, and $(s, k) = \mathsf{Gen}(1^n, \Lambda)$, and output $(pk, sk) = ((pk_0, pk_1, s), (k : sk_0, sk_1))$.

Encryption of a message $m \in \mathcal{M}_{pk_0} \cap \mathcal{M}_{pk_0}$**.** Choose $r_0, r_1 \in \{0,1\}^*$ randomly, compute $c_0 = \mathsf{Enc}^{(0)}_{pk_0}(m, r_0)$, $c_1 = \mathsf{Enc}^{(1)}_{pk_1}(m, r_1)$, and $\pi = \mathsf{Eval}(\Lambda, s, (c_0, c_1), (r_0, r_1)) = H_k(c_0, c_1)$, and output (c_0, c_1, π).

Decryption of a ciphertext (c_0, c_1, π)**.** Output $\mathsf{Dec}^{(0)}_{sk_0}(c_0)$ if $\mathsf{PEval}(\Lambda, k, c_0, c_1) = H_k(c_0, c_1) = \pi$ and otherwise output $\perp$.

Sometimes the keys of the two cryptosystems and/or the ciphertexts may be dependent. We say that the cryptosystems are siblings when the construction works even if this is the case and define this below.

Definition 8. *Two cryptosystems $\mathcal{CS}^{(0)}$ and $\mathcal{CS}^{(1)}$ are siblings if there exists deterministic algorithms $\mathsf{sCSKg}^{(b)}$, $\mathsf{Mall}^{(b)}$ for $b \in \{0,1\}$, such that for every $r, t_0, t_1 \in \{0,1\}^*$, if we define $\mathsf{CSKg}^{(b)}(1^n, (t_0, t_1)) = (pk_b, sk_b)$ and $c_b = \mathsf{Enc}^{(b)}_{pk_b}(m, r)$, then $\mathsf{sCSKg}^{(b)}(pk_{1-b}, t_b) = (pk_b, sk_b)$ and $\mathsf{Mall}^{(b)}(pk_0, pk_1, c_{1-b}, m, t_b) = c_b$.*

If the two cryptosystems in the scheme above are siblings we assume that the same random string (t_0, t_1) is used to generate both key pairs, and we assume that the same random string r is used to form both ciphertexts. We call this the sibling version.

Proposition 2. *If $\mathcal{CS}$ is polynomially indistinguishable, then the cryptosystem above is CCA2-PKE-secure, including the sibling version.*

The difference between the scheme above and that presented by Sahai [20] (based on [16]) is that the non-interactive adaptively zero-knowledge simulation sound proof is replaced by a hash proof. The role of one-time simulation soundness of the non-interactive proof corresponds to the projective hashing lemma (Lemma 1) of the hash proof. The role of the adaptive zero-knowledge simulator corresponds to the private evaluation algorithm and smoothness of the hash proof. The non-interactive proof has one property which has no counterpart in a hash proof; public verifiability.

The Original Cramer-Shoup Scheme. That the original Cramer-Shoup scheme is an optimized instantiation of the sibling version can be seen as follows. Let h be a generator of G_q and let be (pk_b, sk_b) equal to the El Gamal key on the form $((g_b, h), z_b)$, where $g_b = h^{z_b}$ and $z_b \in \mathbb{Z}_q$ is random. This does not change the distribution of h or g_b compared to the original. Note also that an El Gamal ciphertext $(u, e) = (g_b^r, h^r m)$ may be decrypted as $e/u^{1/z_b}$. Then to generate the hash proof parameter on input (g_0, g_1, h), pick $x_0, x_1, y_0, y_1 \in \mathbb{Z}_q$ randomly, define $c = g_0^{x_0} g_1^{x_1}$ and $d = g_0^{y_0} g_1^{y_1}$, and set $(s, k) = ((g_0, g_1, c, d), (x_0, x_1, y_0, y_1))$. The public evaluation algorithm takes (g_0, g_1, c, d) and (u_0, u_1, e, r) as input and outputs $c^r d^{rH(u_0,u_1,e)}$. The private evaluation algorithm takes (u_0, u_1, e), and (x_0, x_1, y_0, y_1) and outputs $u_0^{x_0} u_1^{x_1} (u_0^{y_0} u_1^{y_1})^{H(u_0,u_1,e)}$. This hash proof system is both smooth and universal$_2$ (see [8]). Then define $(pk, sk) = ((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1 : z_0, z_1))$. A ciphertext would have the form $((u_0, e), (u_1, e), v) = ((g_0^r, h^r m), (g_1^r, h^r m), c^r d^{rH((u_0,e),(u_1,e))})$. It is obviously not necessary to send two copies of e, and neither is it necessary to input two copies of e to H or to use both z_0 and z_1 which explains how to get the original Cramer-Shoup scheme.

The Cramer-Shoup Version of the Paillier Scheme. In a similar way it can be seen that the efficient CCA2-secure version of the Paillier [17] cryptosystem presented in [9,3] is also an optimized instantiation of the sibling version. Recall the Paillier

cryptosystem [17]. The key generator CSKg chooses safe primes p and q, computes $N = pq$, and outputs the key pair (p, N). Let $b = 1 + N$. To encrypt a message $m \in \mathbb{Z}_N$ choose $r \in \mathbb{Z}_N^*$ randomly and output $e = \mathsf{Enc}_N(m, r) = b^m r^N \bmod N^2$. To decrypt, compute $m = \mathsf{Dec}_p(e) = ((e^t \bmod N^2) - 1)/N$, where $t = 1 \bmod N$ and $t = 0 \bmod \phi(N)$. A natural variation of this is to let a modified key generator CSKg' also choose a random generator h of the group of $2N$th residues in $\mathbb{Z}_{N^2}^*$ by finding a random element in $\mathbb{Z}_{N^2}^*$ and exponentiate it by $2N$. Then encryption is defined by $\mathsf{Enc}'_N(m, r) = b^m h^r \bmod N^2$, where r is chosen randomly in $[0, N^2]$. This changes the distribution of a ciphertext only negligibly. The modification makes a ciphertext look almost like the second component of an El Gamal ciphertext. We can make decryption independent of the factorization of N by using a further modified key generator CSKg which chooses $z \in [0, N^2]$ randomly, and defines $g = h^z \bmod N^2$. Then the encryption of m is defined by $(u, e) = \mathsf{Enc}''_{(g,h)}(m, r) = (g^r, b^m h^r)$ and decryption of a ciphertext (u, e) by $\mathsf{Dec}''_z(u, e) = e/u^z \bmod N^2$. We can now apply the same argument as is done for the El Gamal instantiation. This would give a symmetrical CCA2-secure version of the Paillier cryptosystem, but not the scheme proposed in [9] as promised.

To get their scheme, we do an asymmetrical combination of the cryptosystems $(\mathsf{CSKg}', \mathsf{Enc}', \mathsf{Dec}')$ and $(\mathsf{CSKg}'', \mathsf{Enc}'', \mathsf{Dec}'')$. These schemes are siblings, since the key generator CSKg'' can be defined using the output of CSKg' and a ciphertext in one scheme then can be translated into one in the other scheme using the secret key of the second scheme. The hash proof is defined by choosing a collision-free hash function H, $x_0, x_1, y_0, y_1 \in [0, N^2]$ randomly and defining $c = g^{x_0} h^{N x_1} \bmod N^2$ and $d = g^{y_0} h^{N y_1} \bmod N^2$, and setting $(s, k) = ((N, g, h, c, d), (x_0, x_1, y_0, y_1))$. The public evaluation algorithm takes (N, g, h, c, d) and (e_0, u_1, e_1, r) as input and outputs $c^r d^{rH(u,e_0,e_1)} \bmod N^2$. The private evaluation algorithm takes (e_0, u_1, e_1) and (x_0, x_1, y_0, y_1) as input and outputs $u_1^{x_0} e_0^{N x_1} (u^{y_0} e_0^{N y_1})^{H(e_0,u_1,e_1)} \bmod N^2$. This hash proof system is both smooth and universal$_2$ (see [9]). Then define the key pair by $(pk, sk) = ((N, g, h, c, d), (x_0, x_1, y_0, y_1 : p, z))$. In the sibling version $e_0 = e_1$, and we can obviously omit one of them. This gives the scheme in [9]. We have cheated a little, in assuming that e_0, u_1, and e_1 are quadratic residues. This can be solved by modifying the encryption algorithms such that they encrypt $m/2 \bmod N$ instead of m, and then consider the output e'_0 or (u'_1, e'_1) as one of two encodings of the same ciphertext $e_0 = (e'_0)^2 \bmod N^2$ or $(u_1, e_1) = ((u'_1)^2 \bmod N^2, (e'_1)^2 \bmod N^2)$, which is then used as above.

Proof (Proposition 2). Denote by $T_b^{(0)}$ the machine that simulates the experiment $\mathsf{Exp}^{\mathsf{cca2-pke}-b}_{\mathcal{CS},A}(n)$ with some adversary $A \in \mathrm{PT}^*$, except that when it computes the challenge ciphertext (c_0, c_1, π) it computes $\pi = \mathsf{PEval}(\Lambda, (c_0, c_1), k)$ instead of $\pi = \mathsf{PEval}(\Lambda, (c_0, c_1), (m, r_0, r_1))$. Furthermore, the decryption oracle in T_1 answers a query $(c_{i,0}, c_{i,1}, \pi_i)$ by $\mathsf{Dec}^{(1)}_{sk_1}(c_{i,1})$ if $H_k(c_{i,0}, c_{i,1}) = \pi_i$ and otherwise by $\perp$.

Claim 1. $|\Pr[\mathsf{Exp}^{\mathsf{cca2-pke}-b}_{\mathcal{CS},A}(n) = 1] - \Pr[T_b^{(0)} = 1]|$ is negligible.

Proof. It follows immediately from the definition of a hash proof system that the first change does not change the distribution at all. Thus, we may concentrate on the second modification.

Define A_{proj} as the adversary that simulates $\Pr[\mathsf{Exp}^{\mathsf{cca2-pke}-b}_{\mathcal{CS},A}(n) = 1]$ and takes part in the experiment of Lemma 1. The simulation is done honestly except that it queries its $\tau_k(\cdot,\cdot)$-oracle instead of computing $H_k(c_{i,0}, c_{i,1}) = \pi_i$, when given a decryption query $(c_{i,0}, c_{i,1}, \pi_i)$, and instead of computing the challenge ciphertext (c_0, c_1, π) honestly it hands (c_0, c_1) to the experiment and uses the value π the experiment returns. Then claim now follows from Lemma 1, since $\mathbf{P}$ is universal_2.

Claim 2. Denote by $T^{(1)}_b$ the machine $T^{(0)}_b$ except that the challenge ciphertext c_{1-b} is defined as $\mathsf{Enc}^{(1-b)}_{pk_{1-b}}(m'_{1-b}, r_{1-b})$ for a randomly chosen $m'_{1-b} \in \mathcal{M}_{pk_{1-b}}$. Then $|\Pr[T^{(0)}_b = 1] - \Pr[T^{(1)}_b = 1]|$ is negligible.

Proof. Define the adversary A_{ind} to be an adversary against the polynomial indistinguishability of $\mathcal{CS}$. It accepts a public key pk_{1-b} as input and simulates $T^{(0)}_b$, running A as a black-box, except for the following modifications. In the sibling version it defines $(pk_b, sk_b) = \mathsf{sCSKg}^{(b)}(pk_{1-b}, t_b)$, for a random $t_b \in \{0,1\}^*$. In the non-sibling version it generates these keys honestly. It interrupts the execution when it is about to compute the challenge ciphertext (c_0, c_1, π). Then it outputs (m_{1-b}, m'_{1-b}), where m'_{1-b} is chosen randomly in $\mathcal{M}_{pk_{1-b}}$. When the polynomial indistinguishability experiment returns c_{1-b} this value is used in the continued simulation of $T^{(0)}_b$. In the sibling version A_{ind} also defines $c_b = \mathsf{Mall}^{(b)}(pk_0, pk_1, c_{1-b}, m_{1-b}, t_b)$. In the non-sibling version it generates these values honestly.

It follows that A_{ind} is identically distributed to $T^{(0)}_b$ or $T^{(1)}_b$ depending on if the polynomial indistinguishability experiment defines c_{1-b} as an encryption of m_{1-b} or m'_{1-b} respectively. Thus, A_{ind} breaks the polynomial indistinguishability of $\mathcal{CS}$ if the claim is false.

Claim 3. Define $T^{(2)}_b$ to be identical to $T^{(1)}_b$ except that the decryption oracle in $T^{(2)}_b$ answers a query $(c_{i,0}, c_{i,1}, \pi_i)$ by $\mathsf{Dec}^{(1-b)}_{sk_{1-b}}(c_{i,1-b})$ if $H_k(c_{i,0}, c_{i,1}) = \pi_i$ and otherwise by $\perp$. Then $|\Pr[T^{(1)}_b = 1] - \Pr[T^{(2)}_b = 1]|$ is negligible.

Proof. This follows from Lemma 1 similarly to the proof of Claim 1.

Claim 4. Denote by $T^{(3)}_b$ the machine that is identical to $T^{(2)}_b$ except that m_b is replaced by a randomly chosen $m'_b \in \mathcal{M}_{pk_b}$. Then $|\Pr[T^{(2)}_b = 1] - \Pr[T^{(3)}_b = 1]|$ is negligible.

Proof. This follows mutatis mutandi from the proof of Claim 2.

Claim 5. Define $T^{(4)}_b$ to be identical to $T^{(3)}_b$ except that the decryption oracle in $T^{(4)}_0$ answers a query $(c_{i,0}, c_{i,1}, \pi_i)$ by $\mathsf{Dec}^{(0)}_{sk_0}(c_{i,0})$ if $H_k(c_{i,0}, c_{i,1}) = \pi_i$ and otherwise by $\perp$. Then $|\Pr[T^{(3)}_b = 1] - \Pr[T^{(4)}_b = 1]|$ is negligible.

Proof. Again, this follows mutatis mutandi from the proof of Claim 1.

Claim 6. Denote by $T_b^{(5)}$ the machine that is identical to $T_b^{(4)}$ except that instead of computing $\pi = H_k(c_0, c_1)$, it simply chooses $\pi \in \Pi$ randomly. Then $|\Pr[T_b^{(4)} = 1] - \Pr[T_b^{(5)} = 1]|$ is negligible.

Proof. Consider an arbitrary fixed instance Λ of the subset membership problem and an arbitrary fixed random string of the experiment. Define a function $f : X \times S \times \Pi \to \{0, 1\}$ as follows. Let $f(x, \alpha(k), \pi)$ simulate $T_b^{(4)}$ except that the input parameters are used in the computation of the challenge ciphertext. Note that although f may not be computable in polynomial time it certainly exists, since $T_b^{(4)}$ outputs $\perp$ if the event E occurs and $\alpha(k)$ determines H_k on L by the projective property of $\mathbf{H}$, so the answers of all queries are determined by $\alpha(k)$. When $k \in K$, $x \in X$, and $\pi \in \Pi$ are randomly chosen, $f(x, \alpha(k), H_k(x))$ is identically distributed to $T_b^{(4)}$ and $f(x, \alpha(k), \pi)$ is identically distributed to $T_b^{(5)}$. The claim now follows from the smoothness of $\mathbf{H}$.

Conclusion of Proof of Proposition. Note that $T_0^{(5)}$ and $T_1^{(5)}$ are identically distributed. The claims above now imply that the proposition holds. □

5 Application to a Mix-Net

The original motivation for this paper was to come up with a non-interactive submission phase in El Gamal based mix-nets. For readers that are not familiar with mix-nets we give an informal description of one such construction that goes back to Sako and Kilian [21].

There are many senders $S_1, \ldots, S_N$ and a small number of mix-servers $M_1, \ldots, M_k$. A secret key z_j and a public key $g_j = h^{z_j}$ are associated with each mix-server, and a joint key $g = \prod_{j=1}^{k} g_j$ is defined. The secret keys z_j may be secret shared to achieve a more robust scheme. To submit a message $m_i \in G_q$ a sender computes an El Gamal ciphertext $(u_{0,i}, e_{0,i}) = (g^{r_i}, h^{r_i} m_i)$, where $r_i \in \mathbb{Z}_q$ is randomly chosen. Then the mix-servers take turns at re-encrypting, using the homomorphic property of El Gamal, and permuting the list of ciphertexts. In other words, for $j = 1, \ldots, k$, M_j computes and publishes $\{(u_{j,i}, e_{j,i})\} = \{(u_{j-1,\pi_j(i)} g^{s_{j,i}}, e_{j-1,\pi_j(i)} h^{s_{j,i}})\}$, where $s_{j,i} \in \mathbb{Z}_q$ is random. Finally, the mix-servers jointly and verifiably decrypt the list $\{(u_{k,i}, e_{k,i})\}$ output by the last mix-server M_k, sort the result and output it. The idea is that due to the transformations computed by the mix-servers the correspondence between the output plaintexts and the input ciphertexts should be hidden. To ensure robustness, the mix-servers also prove knowledge of a witness of the transformation it computes on the list of ciphertexts.

Unfortunately, the above straight-forward construction is completely insecure [18], since a malicious sender S_l may compute its ciphertext as $(u_{0,l}, e_{0,l}) = (u_{0,i}^a, e_{0,i}^a)$ for some random exponent and then identify a matching pair (m, m^a) in the final output. Note that this reveals the message sent by the designated honest sender S_i. Intuitively, what is needed is a non-malleable cryptosystem, but on the other hand

the cryptosystem must be homomorphic for re-encryption to be possible. Formally, what is needed in the overall proof of security of the mix-net is a way to extract the messages submitted by corrupted players without using the secret key of the cryptosystem, as explained in the introduction.

We augment the above to make the cryptosystem used for submission identical to the original Cramer-Shoup scheme. We rename $g_0 = g$, $z_{0,j} = z_j$, and $g_{0,j} = h^{z_{0,j}}$, and then introduce $g_{1,j} = h^{z_{1,j}}$, $g_1 = \prod_{j=1}^{k} g_{1,j}$, $x_{0,j}, x_{1,j}, y_{0,j}, y_{1,j} \in \mathbb{Z}_q$, $c_j = g_0^{x_{0,j}} g_1^{x_{1,j}}$, $d_j = g_0^{y_{0,j}} g_1^{y_{1,j}}$, $c = \prod_{j=1}^{k} c_j$, and $d = \prod_{j=1}^{k} d_j$. This gives a Cramer-Shoup key pair $((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1 : z_0, z_1))$ with distributed secret key.

If the original Cramer-Shoup scheme is used and we only exploit its CCA2-security, then the mix-servers would have to execute some protocol to find the set of valid ciphertexts or randomize those that are invalid as is done in [6]. This requires communication between the mix-servers at least linear in the number of senders.

However, due to the CCA2-PKE-security of the cryptosystem the mix-servers may simply reconstruct the second part of the shared key. This allows each mix-server to identify the valid ciphertexts *without any additional communication*, and form the list of El Gamal ciphertexts consisting of the El Gamal part of each valid ciphertext. Then the mix-net is executed on this list as before.

6 Future Work

In the mix-net application all messages are free-form. This may not be the case in some applications. It is for example not the case in some multi-candidate homomorphic election schemes, e.g., [7], where the submitted messages must be on a specific form to encode a valid candidate. An interesting question is if it is possible to come up with an efficient hash proof system that constrains the set of messages in this way. This would give a very efficient non-interactive submission phase for such election schemes in the standard model.

References

1. M. Abe, R. Cramer, and S. Fehr. Non-interactive distributed-verifier proofs and proving relations among commitments. In *Advances in Cryptology – Asiacrypt 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 206–223. Springer Verlag, 2002.
2. M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In *20th ACM Symposium on the Theory of Computing (STOC)*, pages 103–118. ACM Press, 1988.
3. J. Camensisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *Advances in Cryptology – Crypto 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer Verlag, 2003.
4. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 136–145. IEEE Computer Society Press, 2001. (Full version at Cryptology ePrint Archive, Report 2000/067, http://eprint.iacr.org, October, 2001.).
5. R. Canetti, O. Goldreich, and S. Halevi. The random oracle model revisited. In *30th ACM Symposium on the Theory of Computing (STOC)*, pages 209–218. ACM Press, 1998.
6. R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In *Advances in Cryptology – Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106. Springer Verlag, 1999.

7. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology – Eurocrypt '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118. Springer Verlag, 1997.
8. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology – Crypto '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer Verlag, 1998.
9. R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. http://homepages.cwi.nl/~cramer/, June 1999.
10. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *23rd ACM Symposium on the Theory of Computing (STOC)*, pages 542–552. ACM Press, 1991.
11. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
12. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
13. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
14. J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for np. In *Advances in Cryptology – Eurocrypt 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358. Springer Verlag, 2006.
15. A. Lysyanskaya and C. Peikert. Adaptive security in the threshold setting: From cryptosystems to signature schemes. In *Advances in Cryptology – Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 331–350. Springer Verlag, 2001.
16. M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM Symposium on the Theory of Computing (STOC)*, pages 427–437. ACM Press, 1990.
17. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology – Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer Verlag, 1999.
18. B. Pfitzmann and A. Pfitzmann. How to break the direct RSA-implementation of mixes. In *Advances in Cryptology – Eurocrypt '89*, volume 434 of *Lecture Notes in Computer Science*, pages 373–381. Springer Verlag, 1990.
19. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer Verlag, 1991.
20. A. Sahai. Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 543–553. IEEE Computer Society Press, 1999.
21. K. Sako and J. Killian. Reciept-free mix-type voting scheme. In *Advances in Cryptology – Eurocrypt '95*, volume 921 of *Lecture Notes in Computer Science*, pages 393–403. Springer Verlag, 1995.

A Omitted Proofs

Proof (Lemma 1). Denote by $((x_i, \pi_{x,i}), \pi_i)$ the ith query of A and let E_i be the event that $H_k(x_i, \pi_{x,i}) = \pi_i$, $x_i \in X \setminus L$, and $i \leq i_l$ or $x_i \neq x$. Condition on arbitrary fixed values of (x, π_x), $\pi = H_k(x, \pi_x)$, and $\alpha(k)$. Then the conditional probability of the event E_i is negligible by universal$_2$-ness of $\mathbf{H}$. Since the fixed values are arbitrary, this holds also without conditioning. Finally, A asks at most a polynomial number of queries and the lemma follows from the union bound. □

Proof (Proposition 1). Conceptually, we follow the proof of Cramer and Shoup, but our proof is somewhat simplified, since we ignore the problem of approximating the hash families by efficiently computable hash families. Denote by $T_b^{(0)}$ the machine that simulates the experiment $\mathsf{Exp}_{\mathcal{CS},A}^{\mathsf{cca2-pke}-b}(n)$ with some adversary $A \in \mathrm{PT}^*$. We change $T_b^{(0)}$ step by step until it is independent of b.

Claim 7. Denote by $T_b^{(1)}$ the machine $T_b^{(0)}$ except that x is chosen randomly in $X \setminus L$. Then $|\Pr[T_b^{(0)} = 1] - \Pr[T_b^{(1)} = 1]|$ is negligible.

Proof. Denote by A_{subset} an algorithm that tries to solve the subset membership problem $\mathbf{M}$. It accepts as input (Λ, x), where x either belongs to $X \setminus L$ or L. It simulates $T_b^{(0)}$ except that it uses the instance Λ and defines the challenge ciphertext $(x, e, \widehat{\pi})$ using x from its input (Λ, x). Note that A_{subset} is identically distributed to $T_b^{(0)}$ or $T_b^{(1)}$ depending on if $x \in L$ or $X \setminus L$. From the hardness of $\mathbf{M}$ follows that $|\Pr[T_b^{(0)} = 1] - \Pr[T_b^{(1)} = 1]|$ is negligible.

Denote by $(\pi_i, e_i, \widehat{\pi}_i)$ the ith query of A to the decryption oracle $\mathsf{Dec}_{sk}(\cdot,\cdot,\cdot)$, and let i_l be the index of the last query before the first output. Denote by $(x, e, \widehat{\pi})$ the challenge ciphertext, and let E be the event that there exists an index i such that A asks a decryption query $(\pi_i, e_i, \widehat{\pi}_i)$ with $\widehat{H}_{\widehat{k}}(x_i, e_i) = \widehat{\pi}_i$, $x_i \in X \setminus L$, and $i \leq i_l$ or $x_i \neq x$.

Claim 8. Denote by $T_b^{(2)}$ the machine $T_b^{(1)}$ except that it halts with output 0 if the event E occurs. Then $|\Pr[T_b^{(1)} = 1] - \Pr[T_b^{(2)} = 1]|$ is negligible.

Proof. Define A_{proj} as the machine that simulates $T_b^{(1)}$ and that takes part in the experiment of Lemma 1. The simulation is done honestly except that whenever $T_b^{(1)}$ needs to compute $\mathsf{PEval}_1(\Lambda, \widehat{k}, x_i, e_i) = \widehat{H}_{\widehat{k}}(x_i, e_i)$ it simply queries the $\tau_{\widehat{k}}(\cdot,\cdot)$ oracle with (x_i, e_i), and instead of computing $\mathsf{Eval}_1(\Lambda, \widehat{s}, x, w, e) = \widehat{H}_{\widehat{k}}(x, e) = \widehat{\pi}$ it outputs (x, e) and waits for $\widehat{H}_{\widehat{k}}(x, e)$ from the experiment. The lemma then implies that $\Pr[E]$ is negligible, which in turn implies the claim.

Note that the lemma can be applied despite that the experiment reveals $\widehat{k}$, since all queries asked by A are asked before *this happens. This observation is the only essential change to the original proof.*

Claim 9. Denote by $T_b^{(3)}$ the machine $T_b^{(2)}$ except that $\widehat{\pi}$ in the challenge ciphertext $(x, e, \widehat{\pi})$ is chosen randomly in Π. Then $|\Pr[T_b^{(2)} = 1] - \Pr[T_b^{(3)} = 1]|$ is negligible.

Proof. This follows from the smoothness of the hash proof similarly as in the proof of Claim 6 in the proof of Proposition 2.

Conclusion of Proof of the Proposition. To conclude the proof of the proposition we simply note that the distributions of $T_0^{(3)}$ and $T_1^{(3)}$ are identical. The claims above now imply that $|\Pr[T_0^{(0)} = 1] - \Pr[T_1^{(0)} = 1]|$ is negligible. □