Simplified Submission of Inputs to Cryptographic Protocols*

Douglas Wikström

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

Abstract. Consider an electronic election scheme implemented using a mix-net; a large number of voters submit their votes and then a smaller number of servers compute the result. The mix-net accepts an encrypted vote from each voter and outputs the set of votes in sorted order without revealing the permutation used. To ensure a fair election, the votes of corrupt voters should be independent of the votes of honest voters, i.e., some type of non-malleability or plaintext awareness is needed. However, for efficiency reasons the servers typically expect inputs from some homomorphic cryptosystem, which is inherently malleable.

In this paper we consider the problem of how non-malleability can be guaranteed in the submission phase and still allow the servers to start their computation with ciphertexts of the appropriate homomorphic cryptosystem. This can clearly be achieved using general techniques, but we would like a solution which is: (1) provably secure under standard assumptions, (2) non-interactive for the submitting parties, (3) very efficient for all parties in terms of computation and communication.

We give the first solution to this problem which has all these properties. Our solution is surprisingly simple and can be based on various Cramer-Shoup cryptosystems. To capture its security properties we introduce a variation of CCA2-security.

A byproduct of our results, of independent interest, is a re-interpretation of the efficient concrete Cramer-Shoup cryptosystems in terms of the Naor-Yung double-ciphertext paradigm.

1 Introduction

Mix-Nets. A mix-net is a cryptographic protocol executed by N senders and k mix-servers, where typically N is quite large and k is fairly small, e.g., $N = 10^4$ and $k = 10$. The functionality implemented by a mix-net corresponds to a trusted party that collects inputs from the senders and then outputs the inputs in sorted order. The main application of mix-nets is for electronic elections. All known efficient robust mix-nets exploit the homomorphic properties of cryptosystems such as the El Gamal cryptosystem [14] in an essential way. A problem with using a homomorphic cryptosystem in the submission phase is that corrupted senders can submit inputs that are related to those of honest senders.

Formally, the proof of security fails. When using the simulation paradigm, e.g., universally composable security [5], a mix-net is said to be secure if for every adversary attacking the mix-net there is an ideal adversary (simulator), typically running the adversary as a black-box, attacking the ideal mix-net (the trusted party mentioned above) such that no environment can tell the two models apart. The simulator does not know the inputs of the honest parties and replaces them in its simulation, e.g., by random messages. It must also extract the inputs of corrupted parties in its simulation and hand them to the ideal mix-net, since otherwise these inputs would be missing in the output from the ideal mix-net and the environment could trivially distinguish the two models. Any successful adversary must result in a successful attack against the underlying cryptosystem, which means that the simulator can not use the secret key of the cryptosystem to extract the inputs of corrupt senders.

* A preliminary version of this paper had the title "CCA2-Security Under Partial Key Exposure and the Cramer-Shoup Cryptosystems". The current version has been extensively rewritten to improve the exposition and to correct a flaw in the analysis of the generic scheme.

General Case. More generally, consider a cryptographic protocol that start with a submission phase where many parties submit ciphertexts formed 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. Typically, efficient protocols exploit the algebraic structure of the cryptosystem, e.g., homomorphic properties. 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 unrelated 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. 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 only a polynomially indistinguishable [15] cryptosystem.

The Submission Problem. The submission problem is how to find a submission scheme such that: (a) the inputs of corrupted parties can be extracted by the simulator without using the secret key, and (b) the result is a list of ciphertexts on the form expected by the servers computing the result. These requirements are essential to allow use of the submission scheme as a prefix to the main protocol, but there are also several natural additional properties that we can look for, or even require, in the submission phase.

1. The solution should be provably secure under standard assumptions in the plain model without any random oracle model or generic groups.
2. The submission of an input should be non-interactive for a submitting party and interaction between the servers should be limited, e.g., independent of the number of senders.
3. The communication and computational complexity of the submitting parties should not depend on the number of servers.

1.1 Previous Work

Informally, we may view any solution to the problem as a form of proof of knowledge of the encrypted plaintext, since any solution must allow the simulator to extract the submitted plaintext without knowledge of the secret key of the polynomially indistinguishable cryptosystem. We classify the solutions in the literature and some minor extensions as follows:

1. An *interactive* proof of knowledge [16] is used, either with a straight-line extractor in the public key setting using the Naor and Yung [19] double-ciphertext trick, or with a rewinding extractor.
2. A non-interactive proof of knowledge using *general techniques* [2] is used. This is *not efficient* for either the submitter or the servers, even using the recent techniques of Groth et al. [17]. Note that this is essentially the construction of a CCA2-secure cryptosystems under general assumptions given by Sahai [23] based on the Naor-Yung double-ciphertext trick, but starting from a concrete polynomially indistinguishable cryptosystem with useful structure.
3. A non-interactive (for the submitter) proof of knowledge based on verifiable secret sharing is used, for example using techniques from Abe, Cramer, and Fehr [1]. Then the *computational and communication complexity of the submitting party grows linearly* with the number of servers, since each server must receive an encrypted secret share.

4. A non-interactive proof of knowledge in the *random oracle model* is used, either using the Naor and Yung trick, or with rewinding. Such solutions are typically efficient, but unfortunately only heuristically secure [6]. Note that the cryptosystems CCA2-secure in the random oracle model given by Shoup and Gennaro [25] may be viewed as instantiations of this solution.
5. An arbitrary CCA2-secure cryptosystem is used and ciphertexts are translated into suitable polynomially indistinguishable ciphertexts using *general multiparty computation* techniques.
6. A special CCA2-secure cryptosystem such that a ciphertext can be transformed more easily into a new ciphertext for the basic polynomially indistinguishable scheme needed by the servers is used. We list the solutions of this type we are aware of below.
   (a) Canetti and Goldwasser [7] and Lysyanskaya and Peikert [18] have given CCA2-secure cryptosystems with distributed decryption which allows transforming ciphertexts into ciphertexts for well known polynomially indistinguishable cryptosystems. These either involve *interaction*, *expensive setup assumptions*, or only work for a *small number of servers*.
   (b) Boneh, Boyen, and Halevi [3] give a CCA2-secure cryptosystem with distributed decryption that may be viewed as containing a polynomially indistinguishable cryptosystem, but its security is based on a *non-standard complexity assumption* based on pairings.
   (c) Cramer, Damgård, and Ishai [8] present a solution based on distributed pseudo random functions and share conversion that is reasonably efficient for a *small number of servers*.

To summarize, there are numerous solutions to the submission problem which satisfies properties (a) and (b), but no such solution has the properties (1)-(3) listed above.

1.2 Our Contribution

The Idea. Recall the original construction of Cramer and Shoup [10]. 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 $(u_0, u_1, e, v) = (g_0^r, g_1^r, h^r m, c^r d^{r H(u_0, u_1, e)})$. To decrypt a tuple $(u_0, u_1, e, v) \in G_q^4$ using the secret key sk the decryption algorithm first checks the validity of the ciphertext by testing if $u_0^{x_0} u_1^{x_1} (u_0^{y_0} u_1^{y_1})^{H(u_0, u_1, e)} = v$. 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 [14] 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 solves the submission problem.

Our Results. To allow us to generalize the above observation and identify a class of cryptosystems for which it applies, we introduce the notion of an *augmented cryptosystem* which contains another cryptosystem with useful structure as a component. We also introduce a strengthened variation of CCA2-security, where the first part of the secret key is given to the adversary *before* it guesses the plaintext of the challenge ciphertext. Then we observe that the

generic scheme of Cramer and Shoup [11] satisfies this stronger definition. Unfortunately, this result does not carry over directly to the most efficient Cramer-Shoup schemes used in practice based on the El Gamal [14] and Paillier [20] cryptosystems. Instead, of proving the security of each case separately, we describe an alternative generic scheme based on a combination of the constructions under general assumptions in Naor and Yung [19] and Sahai [23], and the generic Cramer and Shoup [11] scheme, from which these and hopefully future similar results follow directly. Finally, we illustrate the usefulness of our definition by applying it to mix-nets.

When using our solution, no new inputs can be accepted after part of the secret key is revealed. This is a minor drawback in our setting, since we have a large number of submitting parties and executions of the underlying protocol are infrequent. When a new session is executed the servers simply generate a new key. However, it may be useful to use the same cryptosystem in the underlying protocol. Thus, our definitions require that the revealed part of the secret key can be replaced by a freshly generated first part of the secret key. This also allows using several independent first parts of the secret key that may be revealed sequentially, i.e., inputs can be processed in batches and then input to the same underlying protocol. We remark that for general threshold decryption, e.g. [7], our approach is not reasonable, since users requesting a decryption expect the result immediately.

We stress that our results differ from those of Elkind and Sahai [13]. They propose a unifying theory of how to construct CCA2-secure cryptosystems. We simply identify an alternative generic scheme to prove and present our observations about the submission problem in a general way. However, a byproduct of our result is a clear explanation of the relation between the efficient Cramer-Shoup schemes and the Naor-Yung paradigm.

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 $\mathcal{M}_{pk}$, $\mathcal{C}_{pk}$, and $\mathcal{R}_{pk}$ the plaintext space, the ciphertext space, and the randomness space respectively.

2 Augmented Cryptosystems

Keeping our observation about the original Cramer-Shoup cryptosystem in mind, we formalize a general class of CCA2-secure cryptosystems that given part of the secret key allow conversion of a ciphertext into a ciphertext of another related cryptosystem. In applications, the second cryptosystem typically has special properties, e.g., it is homomorphic, that are exploited by the cryptographic protocol. We introduce the following definition.

Definition 1 (Augmented Cryptosystem). *A cryptosystem* $\mathcal{CS} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ *is an augmentation of another cryptosystem* $\mathcal{CS}' = (\mathsf{Kg}', \mathsf{Enc}', \mathsf{Dec}')$ *if there exists an augmentation algorithm* $\mathsf{Aug} \in \mathrm{PPT}$ *and a stripping algorithm* $\mathsf{Strip} \in \mathrm{PT}$ *with the following properties, where we define the random variables* $(pk, sk) = ((pk_1 : pk_2), (sk_1 : sk_2)) = \mathsf{Kg}(1^n)$ *and* $(pk', sk') = \mathsf{Kg}'(1^n)$.

1. *The pair* (pk_2, sk_2) *is identically distributed to* (pk', sk').

2. *For every fixed* (pk_2^*, sk_2^*) *in the support of* (pk_2, sk_2)*, the distribution of* $\mathsf{Aug}(pk_2^*)$ *is identical to the distribution of* pk_1 *conditioned on* $(pk_2, sk_2) = (pk_2^*, sk_2^*)$.
3. *For every* $m \in \mathcal{M}_{pk}$*:* $\mathsf{Dec}'_{sk_2}(\mathsf{Strip}_{sk_1}(\mathsf{Enc}_{pk}(m))) = m$.

Clearly, any cryptosystem can be viewed as a trivial augmentation of itself, and if it is CCA2-secure then the trivial augmentation is also submission secure as defined below. We are interested in non-trivial augmentations where $\mathcal{CS}'$ has structural properties.

Some readers may find it tempting to use a definition that mirrors the Cramer-Shoup cryptosystem more closely to avoid the existence of trivial augmentations, i.e., one could explicitly require that it is possible to check the "validity" of a ciphertext using sk_1. We remind those readers that for most cryptographic notions there are trivial instances, e.g., the identity map is a cryptosystem, and we see no reason to impose unnecessary conditions.

2.1 Submission Security of Augmented Cryptosystems

Before we define submission security of augmented cryptosystem, we recall the game considered in the definition of CCA2-security [19,12,22]. 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 $1/2$.

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.

Experiment 1 (Submission Security, $\mathsf{Exp}^{\mathsf{sub}-b}_{\mathcal{CS},A}(n)$**).**

$$
\begin{aligned}
(pk, (sk_1 : sk_2)) &\leftarrow \mathsf{Kg}(1^n) \\
(m_0, m_1, state_1) &\leftarrow A^{\mathsf{Dec}_{sk}(\cdot)}(\mathsf{choose}, pk) \\
c &\leftarrow \mathsf{Enc}_{pk}(m_b) \\
state_2 &\leftarrow A^{\mathsf{Dec}_{sk}(\cdot)}(\mathsf{morequeries}, state_1, c) \\
d &\leftarrow A(\mathsf{guess}, state_2, sk_1)
\end{aligned}
$$

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

Definition 2 (Submission Security). *An augmentation* $\mathcal{CS}$ *of* $\mathcal{CS}'$ *is said to be submission secure if for all adversaries* $A \in \mathrm{PT}^*$*:* $|\Pr[\mathsf{Exp}^{\mathsf{sub}-0}_{\mathcal{CS},A}(n) = 1] - \Pr[\mathsf{Exp}^{\mathsf{sub}-1}_{\mathcal{CS},A}(n) = 1]|$ *is negligible.*

3 Generic Submission Secure Augmented Cryptosystems

The fact that the generic CCA2-secure cryptosystem of Cramer and Shoup is submission secure if we view it as an augmentation of a basic polynomially indistinguishable cryptosystem 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 [11] for details). Then we introduce an alternative generic cryptosystem and explain how the efficient Cramer-Shoup schemes based on the El Gamal cryptosystem [14] and the Paillier cryptosystem [20] are instantiations of this.

3.1 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 L or from $X \setminus L$. To be useful in cryptography we also need some algorithms that allow us to sample instances and elements, and check for membership in X.

Definition 3. *A subset membership problem $\mathbb{M}$ 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 4. *Let $\mathbb{M}$ 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 $\mathbb{M}$ is hard if for all $A \in \mathrm{PT}^*$: $|\Pr[A(\Lambda, x) = 1] - \Pr[A(\Lambda, x') = 1]|$ is negligible.*

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

Definition 5. *A projective hash family $\mathbb{H} = (H, K, X, L, \Pi, S, \alpha)$ consists of finite non-empty sets K, X, $L \subsetneq X$, Π, and S, a function $\alpha : K \to S$, and a collection of hash functions $H = (H_k : X \times \Pi \to \Pi)_{k\in K}$, where $\alpha(k)$ determines H_k on $L \times \Pi$.*

Definition 6. *Let $\mathbb{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and let $k \in K$ be random. Then $\mathbb{H}$ is universal$_2$ if for all $s \in S$, $x, x' \in X$ with $x \notin L \cup \{x'\}$, and $\pi_x, \pi'_x, \pi, \pi' \in \Pi$, $\Pr_k[H_k(x, \pi_x) = \pi \wedge H_k(x', \pi'_x) = \pi' \wedge \alpha(k) = s]$ is negligible.*

The following definition and lemma are stated informally in [11].

Experiment 2 (Computationally Universal$_2$, $\mathsf{Exp}^{\mathsf{cuni}_2}_{\mathbb{H},A}(n)$). Let τ_k be the predicate defined by $\tau_k((x, \pi_x), \pi) \iff H_k(x, \pi_x) = \pi$, and consider the following experiment.

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

Denote by $((x_i, \pi_{x,i}), \pi_i)$ the ith query to τ_k, and let i_l be the index of the last query before the first output. If 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 $i \le i_l$ or $x_i \neq x$, then output 1 and otherwise 0.

Definition 7. *A projective hash family $\mathbb{H}$ is computationally universal$_2$ if for every $A \in \mathrm{PT}^*$, $\Pr[\mathsf{Exp}^{\mathsf{cuni}_2}_{\mathbb{H},A}(n) = 1]$ is negligible.*

Lemma 1. *If a projective hash family $\mathbb{H}$ is universal$_2$, then it is computationally universal$_2$.*

Definition 8. *Let $\mathbb{H} = (H, K, X, L, \Pi, S, \alpha)$ be a projective hash family, and let $k \in K$, $x \in X \setminus L$, and $\pi \in X$ be random. Then $\mathbb{H}$ 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, in the unconditional case, to a special case of what Cramer and Shoup [11] call "extended strongly (smooth, universal$_2$)" hash proof system.

Definition 9. *A (smooth, (computational) universal$_2$) hash proof system $\mathbb{P}$ for a subset membership problem $\mathbb{M}$ associates with each instance $\Lambda[X, L, W, R]$ a (smooth, (computationally) universal$_2$) projective hash family $\mathbb{H} = (H, K, X, L, \Pi, S, \alpha)$, 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 $(x, w) \in R$ and $\pi_x \in \Pi$, 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.*

3.2 The Generic Scheme of Cramer and Shoup

Given the definitions above it is not hard to describe the generic cryptosystem of Cramer and Shoup [11]. Let $\mathbb{M}$ be a hard subset membership problem, such that Π can be fitted with a group operation for any instance Λ, let $\mathbb{P}_0 = (\mathsf{Gen}_0, \mathsf{PEval}_0, \mathsf{Eval}_0, \mathsf{Mem}_0)$ be a smooth hash proof system for $\mathbb{M}$, and let $\mathbb{P}_1 = (\mathsf{Gen}_1, \mathsf{PEval}_1, \mathsf{Eval}_1, \mathsf{Mem}_1)$ be a computationally universal$_2$ hash proof system for $\mathbb{M}$.

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 $m = e - \mathsf{PEval}_0(\Lambda, k, x) = e - H_k(x)$, 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, except that we have introduced a comma in the notation of the secret key to distinguish the two parts as needed in the definition of submission security. For ease of presentation we do not explicitly put a comma in the public key. Note that the scheme is an augmentation of the polynomially indistinguishable cryptosystem that has public key (Λ, s) and secret key k, and where a message $m \in \Pi$ is encrypted as (x, e), where $e = \mathsf{Eval}_0(\Lambda, s, x, w) + m$, and a ciphertext (x, e) is decrypted by computing $e - \mathsf{PEval}_0(\Lambda, k, x)$.

Cramer and Shoup prove (see Theorem 1 in [11]) that the above cryptosystem is CCA2-secure under the assumption that $\mathbb{M}$ is hard. We show the slightly stronger result.

Proposition 1. *If $\mathbb{M}$ is a hard subset membership problem, then the generic Cramer-Shoup cryptosystem above is submission 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 submission security experiment, since no queries are asked after this point.

3.3 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 [19], Cramer and Shoup [10], and Sahai [23]. This allows a single proof of security of the most commonly used efficient Cramer-Shoup cryptosystems and exposes an interesting relation between these approaches.

Denote by $\mathcal{CS}^0 = (\mathsf{Kg}^0, \mathsf{Enc}^0, \mathsf{Dec}^0)$ and $\mathcal{CS}^1 = (\mathsf{Kg}^1, \mathsf{Enc}^1, \mathsf{Dec}^1)$ cryptosystems. Denote by $\mathbb{M} = ((I_n)_{n\in\mathbb{N}}, \mathsf{Ins}, \mathsf{Sam}, \mathsf{Mem})$ the subset membership problem defined as follows. To sample I_n, compute $(pk_0, sk_0) = \mathsf{Kg}^0(1^n)$ and $(pk_1, sk_1) = \mathsf{Kg}^1(1^n)$, and output (pk_0, pk_1). To define an instance $\Lambda[X, L, W, R]$ from these public keys, let $X = \mathcal{C}_{pk_0} \times \mathcal{C}_{pk_1}$, and let R be the relation of pairs $((c_0, c_1), (m, r_0, r_1))$ in $(\mathcal{C}_{pk_0} \times \mathcal{C}_{pk_1}) \times (\mathcal{M}_{pk_0} \cap \mathcal{M}_{pk_1} \times \mathcal{R}_{pk_0} \times \mathcal{R}_{pk_1})$ such that $c_0 = \mathsf{Enc}^0_{pk_0}(m, r_0) \wedge c_1 = \mathsf{Enc}^1_{pk_1}(m, r_1)$, 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 $\mathbb{P} = (\mathsf{Gen}, \mathsf{PEval}, \mathsf{Eval}, \mathsf{Mem})$ be a computational universal$_2$ hash proof system for $\mathbb{M}$. We then define the cryptosystem as follows.

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

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

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

Note that the result may be viewed as an augmentation of $\mathcal{CS}^0$. Sometimes the keys of the two cryptosystems and/or the ciphertexts may be dependent. We say that $\mathcal{CS}^1$ is a child of $\mathcal{CS}^0$ when this is the case and define this below.

Definition 10 (Child). *Let $\mathcal{CS}^0$ and $\mathcal{CS}^1$ be cryptosystems. Let $t_0, t_1 \in \{0,1\}^*$ be random tapes and define $(pk_0, sk_0) = \mathsf{Kg}^0(1^n, t_0)$, $(pk_1, sk_1) = \mathsf{Kg}^1(1^n, (t_0, t_1))$. We say that $\mathcal{CS}^1$ is a child of $\mathcal{CS}^0$ if there exists algorithms $\mathsf{KeyMall}, \mathsf{Mall} \in \mathrm{PPT}$, and a bit $\beta \in \{0,1\}$ such that:*

1. $\mathsf{KeyMall}(pk_0, t_1) = (pk_1, sk_1)$, $\mathcal{M}_{pk_0} = \mathcal{M}_{pk_1}$, *and* $\mathcal{R}_{pk_0} = \mathcal{R}_{pk_1}$.
2. *If $r \in \mathcal{R}_{pk_0}$ is random, $m \in \mathcal{M}_{pk_0}$, $c_0 = \mathsf{Enc}^0_{pk_0}(m, r)$, and $c_1 = \mathsf{Enc}^1_{pk_1}(m, r)$, then:*
   (a) $(c_0, \mathsf{Mall}(pk_0, pk_1, c_0, m, t_1))$ and (c_0, c_1) are identically distributed, and
   (b) if $m' \in \mathcal{M}_{pk_0}$ is random, then for every $A \in \mathrm{PPT}$ with $c_1' = \mathsf{Mall}(pk_0, pk_1, c_0, m', t_1)$:

$$\left| \Pr\left[A^{\mathsf{Dec}^{\beta}_{sk_\beta}(\cdot)}(pk_0, pk_1, c_0, c_1') = 1 \right] - \Pr\left[A^{\mathsf{Dec}^{\beta}_{sk_\beta}(\cdot)}(pk_0, pk_1, c_0, c_1) = 1 \right] \right| .$$

This simply means that: (1) given the public key pk_0 of $\mathcal{CS}^0$, dependent public and secret keys of $\mathcal{CS}^1$ can be sampled, (2a) given a ciphertext $c_0 = \mathsf{Enc}^0_{pk_0}(m, r)$, a ciphertext $c_1 =$

$\mathsf{Enc}^1_{pk_1}(m,r)$ of the same message m and using the same randomness r can be computed using sk_1 and m, and (2b) when m' is random, the fake ciphertext c'_1 is indistinguishable from $\mathsf{Enc}^1_{pk_1}(m,r)$ even if the adversary is given access to the βth decryption oracle.

We may view any pair $(\mathcal{CS}^0, \mathcal{CS}^1)$ of polynomially indistinguishable cryptosystems as if $\mathcal{CS}^1$ is a child of $\mathcal{CS}^0$ if we change their key generation algorithms and encryption algorithms as follows. The key generator Kg^1 ignores the first part t_0 of its random tape. The random tape r to the encryption algorithm Enc^b is considered as a pair $r = (r_0, r_1)$ with $r_b \in \mathcal{R}_{pk_b}$, and the part r_{1-b} is ignored. This clearly does not change the security properties of the cryptosystems. The algorithms $\mathsf{KeyMall}$ and Mall may then be defined as $\mathsf{KeyMall}(pk_0, t_1) = \mathsf{Kg}^1(t_1)$ and $\mathsf{Mall}(pk_0, pk_1, c_0, m, t_1) = \mathsf{Enc}^1_{pk_1}(m)$. It is clear that properties (1) and (2a) hold, and property (2b) follows immediately from the polynomial indistinguishability of $\mathcal{CS}^1$. Thus, it suffices to consider the child-version of the cryptosystem constructed above.

Proposition 2. *If $\mathcal{CS}^0$ and $\mathcal{CS}^1$ are polynomially indistinguishable, then the child-version of the cryptosystem is submission secure.*

3.4 Two Important Instantiations

We now explain how the two most efficient schemes of Cramer and Shoup may be viewed as direct instantiations of the alternative generic scheme.

The Original Cramer-Shoup Scheme. We first explain informally the relation between the Cramer-Shoup scheme and the Naor-Yung paradigm [19,23]. Start with two copies of the El Gamal cryptosystem with public keys (g, h_0) and (g, h_1), where $h_b = g^{z_b}$. The question is how to construct the proof of equal plaintexts of two ciphertexts $(g^r, h_0^{r_0} m)$ and $(g^r, h_1^{r_1} m)$ of the same message. Cramer and Shoup's solution may be re-interpreted as follows.

1. Switch the roles of g and h_b, i.e., we define (g_b, h) as the bth public key where $g_b = h^{z_b}$. Decryption of (u, v) is then re-defined to be vu^{-1/z_b} and encryption remains unchanged. It is obvious that the cryptosystem remains secure.
2. Reuse the randomness $r \in \mathbb{Z}_q$ when forming two ciphertexts of the same message m, i.e., encrypt m using both public keys as $(u_0, e_0) = (g_0^r, h^r m)$ and $(u_1, e_1) = (g_1^r, h^r m)$. That the result remains secure under the DDH assumption is easy to see and well known.
3. Observe that only the holder of the secret key of the cryptosystem needs to verify the proof of equal plaintexts, and that proving this is equivalent to proving that $\log_{g_0} u_0 = \log_{g_1} u_1$ with e_0 as a label, since $e_0 = e_1$.
4. A proof of equal plaintexts is then done using the ingenious notion of a hash proof.

Surprisingly, we have never heard anybody explain the Cramer-Shoup cryptosystem in this way (we have of course dodged the question of how to construct the hash proof).

We now explain in detail how we may view the original Cramer-Shoup scheme as an instance of the alternative generic construction. 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. An El Gamal ciphertext $(u_b, e_b) = (g_b^r, h^r m)$ is then decrypted as $e_b / u_b^{1/z_b}$. It is not hard to see that this cryptosystem is a child of itself, which explains the b-subscript in our notation. Compared to Definition 10 we set $t_0 = z_0$, $t_1 = z_1$, and define $\mathsf{KeyMall}((g_0, h), z_1) = (h^{z_1}, h)$ and $\mathsf{Mall}((g_0, h), (g_1, h), (u_0, e_0), m', z_1) = ((e_0/m')^{z_1}, e_0)$. It is clear that properties (1) and (2a) of Definition 10 holds. Property (2b) holds with $\beta = 0$ under the decision Diffie-Hellman

assumption, since given a tuple (m, A, B, C), we may define $h = m^s$, $g_0 = h^{z_0}$, $g_1 = A^s$, $(u_0, e_0) = (B^{sz_0}, B^s m) = (g_0^b, h^b m)$, and $(u_1, e_1) = (C^s, e_0)$. Then u_1 is on the form g_1^b or $(e_0/m')^{z_1}$ for a random $m' \in G_q$ depending on if the tuple (m, A, B, C) is on the form (m, m^a, m^b, m^{ab}) or on the form (m, m^a, m^b, m^c) for random $a, b, c \in \mathbb{Z}_q$.

All we need now is a hash proof that $\log_{g_0} u_0 = \log_{g_1} u_1$. To generate the hash proof parameter on input (g_0, h) and (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, h), (g_1, h), and (g_0, g_1, c, d) and $((u_0, e_0), (u_1, e_1), r)$ as input and outputs $c^r d^{rH((u_0,e_0),(u_1,e_1))}$. The private evaluation algorithm takes (u_0, e_0), (u_1, e_1), and (x_0, x_1, y_0, y_1) as input and outputs a proof $u_0^{x_0} u_1^{x_1} (u_0^{y_0} u_1^{y_1})^{H((u_0,e_0)(u_1,e_1))}$. This hash proof system is computationally universal$_2$ (see [10]). The key pair of the concrete scheme is given by $(pk, sk) = ((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1 : z_0))$, and a concrete ciphertext would then have the form $((u_0, e_0), (u_1, e_1), v) = ((g_0^r, h^r m), (g_1^r, h^r m), c^r d^{rH((u_0,e_0),(u_1,e_1))})$.

It is obviously not necessary to send two copies of e_0, and neither is it necessary to input two copies of e_0 to H which explains how to get the original Cramer-Shoup scheme, where two copies of these elements are not present.

Cramer-Shoup Versions of the Paillier Scheme. In a similar way it can be seen that the efficient CCA2-secure version of the Paillier [20] cryptosystem presented in [11,4] is also an optimized instantiation of the child-version.

Recall the Paillier cryptosystem [20]. The key generator Kg chooses safe primes p and q, computes $N = pq$, and outputs the key pair (N, p). 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 Kg' also choose a random generator h of the group of $2N$th residues in $\mathbb{Z}_{N^2}^*$. To sample such an element, choose an element in $\mathbb{Z}_{N^2}^*$ randomly and exponentiate it by $2N$. Then encryption is defined by $\mathsf{Enc}'_{(N,h)}(m, r) = b^m h^r \bmod N^2$, where r is chosen randomly in $[0, N^2]$, and decryption Dec' remains the same. This changes the distribution of ciphertexts 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 Kg'' which chooses $z \in [0, N^2]$ randomly, and defines $g = h^z \bmod N^2$. Then encryption of $m \in \mathbb{Z}_N$ is defined by $(u, e) = \mathsf{Enc}''_{(N,g,h)}(m, r) = (g^r, b^m h^r)$ and decryption of a ciphertext (u, e) is defined by $\mathsf{Dec}''_z(u, e) = ((e^z/u \bmod N^2) - 1)/Nz$ if $e^z/u \bmod N^2$ is on the form b^m and $\perp$ otherwise. This works, since $e^z = b^{mz} u \bmod N^2$, $b^{mz} = 1 + Nzm \bmod N^2$, and with overwhelming probability z and N are relatively prime. 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 using the corresponding hash proof. This can then be optimized in the same way as the El Gamal based scheme above, but it still does not give the cryptosystem in [11] as promised (their scheme has 3 group elements and not 4).

Combine the two cryptosystems above and note that the latter is a child of the former. Define $((N_1, g_1, h_1), z_1) = \mathsf{KeyMall}((N_0, h_0), z_1) = ((N_0, h_0^{z_1}, h_0), z_1)$. This implies $N_0 = N_1$ and $h_0 = h_1$, but we keep the subscripts for clarity. Define $\mathsf{Mall}((N_0, h_0), (N_1, g_1, h_1), e_0, m', z_1) = ((e_0 b^{-m'})^{z_1}, e_0)$. It is clear that properties (1) and (2a) of Definition 10 holds. To see that property (2b) holds with $\beta = 1$ under the composite residuosity assumption, note that we can modify key generation and define $h_1 = g_1^{Nz'}$ and set $h_0 = h_1$, where g_1 is a random $2N$th residue and $z' \in [0, N^2]$ is randomly chosen, with a statistically small change in the distribution

of these elements. Moreover, given z' we can decrypt (u_1, e_1) by computing $e_1 u_1^{-Nz'}$, i.e., the decryption oracle in Definition 10 can be simulated. Finally, given a square $A = h_1^a b^{-m'}$ in $\mathbb{Z}_{N^2}^*$ we may define $e_0 = A^{Nz'} b^m = h_0^a b^m$ and $(u_1, e_1) = (A, e_0) = (g_0^a b^{-m'}, h_1^a b^m)$, and conclude that the distribution of (e_0, u_1, e_1) is identically distributed to $(\mathsf{Enc}'_{(N_0,h_0)}(m,a), \mathsf{Enc}''_{(N_1,g_1,h_1)}(m,a))$ or $(\mathsf{Enc}'_{(N_0,h_0)}(m,a), \mathsf{Mall}((N_0,h_0),(N_1,g_1,h_1),e_0,m',z_1))$ for a random $m' \in \mathbb{Z}_N$, depending on if A is a random $2N$th residue or square. Under the composite residuosity assumption these two distributions are indistinguishable. We now drop the subscripts from $h_0 = h_1$ and g_1.

All we need now is a hash proof that $\log_g u_1 = \log_{h^N} e_0^N$, since this implies that e_0 and (u_1, e_1) encrypt the same message. The hash proof is defined by choosing a collision-free hash function H and $x_0, x_1, y_0, y_1 \in [0, N^2]$ randomly and defining $c = g^{x_0} h^{Nx_1} \bmod N^2$ and $d = g^{y_0} h^{Ny_1} \bmod N^2$, and setting $(s,k) = ((N,g,h,c,d,H),(x_0,x_1,y_0,y_1))$. The public evaluation algorithm takes (N,g,h,c,d) and $(e_0,(u_1,e_1),m,r)$ as input and outputs the element $c^r d^{rH(e_0,u_1,e_1)} \bmod N^2$. The private evaluation algorithm takes (e_0,u_1,e_1) and (x_0,x_1,y_0,y_1) as input and outputs $u_1^{x_0} e_0^{Nx_1} (u_1^{y_0} e_0^{Ny_1})^{H(e_0,u_1,e_1)} \bmod N^2$. That this is a hash proof can be seen in the same way as for the El Gamal version. This is slightly differently formulated from the original Paillier version and of independent interest, since this verification is simpler to prove in zero-knowledge than the original (a zero-knowledge proof for the original verification is given in [4]). In the original it is verified explicitly that $e_1/u_1^{z_1} = b^m$ for some m. If we do this, then we may in fact simplify the verification to $u_1^{x_0+zNx_1}(u_1^{y_0+zNy_1})^{H(e_0,u_1,e_1)} = u_1^{x_0'+x_1'H(e_0,u_1,e_1)}$ and then it is easy to see that x_0' and x_1' may be taken as random elements in $[0,N^2]$, since it changes the view of the adversary only statistically. This is the construction in [11,4].

To simplify the exposition we cheated a little above in that we have assumed that all elements belong to the subgroup of squares of $\mathbb{Z}_{N^2}^*$, but this problem is easily resolved [11,4].

3.5 Are All CCA2-Secure Cryptosystems Submission Secure?

From our results it may appear that all CCA2-secure cryptosystems are also submission secure, but the following construction shows that this is not the case even if it loosely speaking is based on the Naor-Yung paradigm.

Let $\mathcal{CS} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be any CCA2-secure cryptosystem. We use $\mathcal{CS}$ twice as a polynomially indistinguishable cryptosystem, and once as a hash proof to construct $\mathcal{CS}'$. The key pair is defined by $((pk, pk_0, pk_1), (sk : sk_0, sk_1))$, where $(pk_0, sk_0) = \mathcal{CS}(1^n)$, $(pk_1, sk_1) = \mathcal{CS}(1^n)$, and $(pk, sk) = \mathcal{CS}(1^n)$. To form a ciphertext of a message m, we compute $(c_0, c_1, c) = (\mathcal{CS}_{pk_0}(m, r_0), \mathcal{CS}_{pk_1}(m, r_1), \mathcal{CS}_{pk}((m, r_0, r_1), r))$. To decrypt a ciphertext (c_0, c_1, c), we compute $\mathsf{Dec}_{sk}(c) = (m, r_0, r_1)$ and verify that $(c_0, c_1) = (\mathcal{CS}_{pk_0}(m, r_0), \mathcal{CS}_{pk_1}(m, r_1))$. If so, then output m and otherwise output $\perp$. It follows from the CCA2-security of $\mathcal{CS}$ that this cryptosystem is CCA2-secure, but it is clearly not submission secure. This is explained by the lack of projectivity in our "secret key non-interactive computational zero-knowledge perfectly sound proof" c. Our "proof" can be simulated by encrypting a random message, but the simulated proof only remains indistinguishable from a real proof until the secret key of the proof system is revealed. A hash proof on the other hand can not only be simulated without the witness using a secret parameter. It can actually be *computed*.

4 Application to a Mix-Net

The original motivation for this paper was to come up with a practical 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 a construction that goes back to Sako and Kilian [24].

There are many senders $S_1, \ldots, S_N$ and a small number of mix-servers $M_1, \ldots, M_k$, e.g., $N = 10^4$ and $k = 10$. A secret key z_j and a public key (g_j, h), with $g_j = h^{z_j}$, are associated with each mix-server, and a joint key $g = \prod_{j=1}^{k} g_j$ is defined. The secret keys z_j are also verifiably secret shared to the other mix-servers. 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 correctness of the transformation it computes on the list of ciphertexts (in fact they must prove *knowledge* of a witness of the transformation as pointed out in [27]).

Unfortunately, the above straight-forward construction is completely insecure [21], 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 a and then identify a matching pair (m, m^a) in the final output. This reveals the message sent by the 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 (see [26,27,29]) is a way to extract the messages submitted by corrupted players without using the secret key of the cryptosystem, as explained in the introduction. In previous work this is either solved heuristically, or as in the cited works a proof of knowledge is used explicitly.

We augment the above to make the cryptosystem used for submission identical to the Cramer-Shoup scheme. We rename $g_0 = g$, $z_{0,j} = z_j$, and $g_{0,j} = h^{z_{0,j}}$, and then introduce $z_1, x_0, x_1, y_0, y_1 \in \mathbb{Z}_q$, $g_1 = h^{z_1}$, $c_j = g_0^{x_0} g_1^{x_1}$, $d = g_0^{y_0} g_1^{y_1}$, where z_1, x_0, x_1, y_0, y_1 are secret shared among the mix-servers. This gives a Cramer-Shoup key pair with distributed secret key on the form $((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1 : z_0))$. Due to the submission security of the cryptosystem the mix-servers may simply reconstruct the first part (x_0, x_1, y_0, y_1) of the shared key before staring the mixing process. 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.

A proof of security of the above mix-net is beyond the scope of this paper. However, the construction and the proof of security in [27] are easily adapted to use the above Cramer-Shoup based submission method. In fact, the proof is simplified. Instead of having extraction of inputs from corrupt parties as a separate step, this comes for free by use of the decryption oracle, and security is reduced to the submission security of the Cramer-Shoup cryptosystem.

5 Future Work

In the mix-net application, all messages are free-form. This may not be the case in other applications. It is for example not the case in multi-candidate homomorphic election schemes, e.g., [9], where the submitted messages must be of a special form to encode a valid candidate. It is an interesting question 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. D. Boneh, X. Boyen, and S. Halevi. Chosen ciphertext secure public key threshold encryption without random oracles. In *Topics in Cryptology - CT-RSA 2006, The Cryptographers' Track at the RSA Conference 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 226–243. Springer Verlag, 2006.
4. 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.
5. 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.).
6. 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.
7. 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.
8. R. Cramer, I. Damgård, and Y. Ishai. Share conversion, pseudorandom secret-sharing and applications to secure computation. In *2nd Theory of Cryptography Conference (TCC)*, volume 3378 of *Lecture Notes in Computer Science*, pages 342–362. Springer Verlag, 2005.
9. 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.
10. 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.
11. 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.
12. 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.
13. E. Elkind and A. Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/042, 2002. http://eprint.iacr.org/.
14. 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.
15. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
16. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
17. 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.
18. 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.
19. 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.
20. 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.
21. 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.
22. 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.

23. 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.
24. 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.
25. V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In *Advances in Cryptology – Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 1–16. Springer Verlag, 1998.
26. D. Wikström. A universally composable mix-net. In *1st Theory of Cryptography Conference (TCC)*, volume 2951 of *Lecture Notes in Computer Science*, pages 315–335. Springer Verlag, 2004.
27. D. Wikström. A sender verifiable mix-net and a new proof of a shuffle. In *Advances in Cryptology – Asiacrypt 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 273–292. Springer Verlag, 2005. (Full version [28]).
28. D. Wikström. A sender verifiable mix-net and a new proof of a shuffle. Cryptology ePrint Archive, Report 2004/137, 2005. http://eprint.iacr.org/.
29. Douglas Wikström and Jens Groth. An adaptively secure mix-net without erasures. In *33rd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 4052 of *Lecture Notes in Computer Science*, pages 276–287. Springer Verlag, 2006.

A 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 universiality$_2$ of $\mathbb{H}$. 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 $\mathsf{Exp}^{\mathsf{sub}-b}_{\mathcal{CS},A}(n)$ with some adversary $A \in \mathrm{PT}^*$, except that when computing the challenge ciphertext $(x, e, \widehat{\pi})$, the two hash proofs π and $\widehat{\pi}$ are computed as $\pi = \mathsf{PEval}_0(\Lambda, k, x) = H_k(x)$ and $\widehat{\pi} = \mathsf{PEval}_1(\Lambda, \widehat{k}, x, e) = \widehat{H}_{\widehat{k}}(x, e)$. By the projectivity of hash proofs this does not change the distribution of the experiment.

We now change $T_b^{(0)}$ step by step until it is independent of b.

Claim 1. 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_{mem} an algorithm that tries to solve the subset membership problem $\mathbb{M}$. 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_{mem} 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 $\mathbb{M}$ 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 2. $\Pr[E]$ is negligible.

Proof. Define A_{uni} as the machine that simulates $T_b^{(1)}$ and that takes part in Experiment 2. The simulation is done honestly except that whenever $T_b^{(1)}$ needs to check a hash proof as $\mathsf{PEval}_1(\Lambda, \widehat{k}, x_i, e_i) = \widehat{H}_{\widehat{k}}(x_i, e_i)$ it simply queries the $\tau_{\widehat{k}}(\cdot,\cdot)$ oracle with (x_i, e_i) instead, and when it needs to compute $\mathsf{Eval}_1(\Lambda, \widehat{s}, x, w, e) = \widehat{H}_{\widehat{k}}(x, e) = \widehat{\pi}$ it outputs (x, e) and waits for $\widehat{H}_{\widehat{k}}(x, e)$ from the experiment instead. The computational universal$_2$ property then implies the lemma.

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.*

Denote by $T_b^{(2)}$ the machine $T_b^{(1)}$, except that it outputs $\perp$ if the event E occurs. The machine $T_b^{(2)}$ may not be efficient, but this does not matter since the remainder of the argument is statistical.

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

Proof. Consider an arbitrary fixed instance Λ of the subset membership problem and an arbitrary fixed random string of the experiment conditioned on the event $\bar{E}$. Define a function $f : X \times S \times \Pi \to \{0, 1\}$ as follows. Let $f(x, \alpha(k), \pi)$ simulate $T_b^{(2)}$ except that the input parameters are used in the computation of the challenge ciphertext. Note that f exists, since $T_b^{(2)}$ outputs $\perp$ if the event E occurs and $\alpha(k)$ determines H_k on L by the projective property of $\mathbb{H}$, 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 $T_b^{(2)}$ and $f(x, \alpha(k), \pi)$ is identically distributed to $T_b^{(3)}$. The claim now follows from the smoothness of $\mathbb{P}$.

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. □

Proof (Proposition 2). Denote by $T_b^{(0)}$ the machine that simulates the experiment $\mathsf{Exp}^{\mathsf{sub}-b}_{\mathcal{CS},A}(n)$ 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 $\pi = \mathsf{Eval}(\Lambda, (c_0, c_1), (m, r))$.

Claim 4. $\Pr[\mathsf{Exp}^{\mathsf{sub}-b}_{\mathcal{CS},A}(n) = 1] = \Pr[T_b^{(0)} = 1]$.

Proof. Follows immediately from the definition of a hash proof system, since a proof of a true statement computed using the secret parameter is identical to one computed using a witness.

Claim 5. Denote by $T_b^{(1)}$ the machine $T_b^{(0)}$ except that the pair (c_0, c_1) is replaced by the pair $(\mathsf{Enc}^0_{pk_0}(m_b, r), \mathsf{Mall}(pk_0, pk_1, c_0, m'_b, t_1))$, where the message $m'_b \in \mathcal{M}_{pk_0}$ is randomly chosen. Then $|\Pr[T_b^{(0)} = 1] - \Pr[T_b^{(1)} = 1]|$ is negligible.

Proof. If $\mathcal{CS}^1$ is a child of $\mathcal{CS}^0$ with $\beta = 0$, then the claim follows from Property (2b) of Definition 10.

Consider now the case where $\mathcal{CS}^1$ is a child of $\mathcal{CS}^0$ with $\beta = 1$. Suppose the decryption oracle in $T_b^{(0)}$ (or $T_b^{(1)}$) 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 $\perp$. This only changes the distribution of $T_b^{(0)}$ (or $T_b^{(1)}$) if a query satisfies $H_k(c_{i,0}, c_{i,1}) = \pi_i$ and $\mathsf{Dec}^1_{sk_1}(c_{i,1}) \neq \mathsf{Dec}^0_{sk_0}(c_{i,0})$. Let us call such queries bad.

To see that a bad query is asked with negligible probability, consider the following. Define an adversary A_{uni} that simulates $T_b^{(0)}$ (or $T_b^{(1)}$) and takes part in the computational universal$_2$ experiment, i.e., Experiment 2. The simulation only changes in that the $\tau_k(\cdot,\cdot)$-oracle is queried 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. It then follows from $\mathbb{P}$'s computational universiality$_2$ property that a bad query is asked with negligible probability. Finally, it follows from Property (2b) of Definition 10 that with these changes $|\Pr[T_b^{(0)} = 1] - \Pr[T_b^{(1)} = 1]|$ is negligible.

Claim 6. Denote by $T_b^{(2)}$ the machine $T_b^{(1)}$ except that the pair (c_0, c_1) is replaced by the pair $(\mathsf{Enc}^0_{pk_0}(m'_b, r), \mathsf{Mall}(pk_0, pk_1, c_0, m'_b, t_1))$, where the message $m'_b \in \mathcal{M}_{pk_0}$ is randomly chosen. Then $|\Pr[T_b^{(1)} = 1] - \Pr[T_b^{(2)} = 1]|$ is negligible.

Proof. Define the adversary A_{ind} to be an adversary against the polynomial indistinguishability of $\mathcal{CS}^0$. It accepts a public key pk_0 as input and simulates $T_b^{(1)}$, running A as a black-box. It interrupts the execution of the experiment when it is about to simulate the computation of the challenge ciphertext (c_0, c_1, π). Then it outputs (m_b, m'_b), where m'_b is chosen randomly in $\mathcal{M}_{pk_0}$. The polynomial indistinguishability experiment then returns c_0. Then c_1 is defined by $c_1 = \mathsf{Mall}(pk_0, pk_1, c_0, m'_b, t_1)$. It follows that A_{ind} is identically distributed to $T_b^{(1)}$ or $T_b^{(2)}$ when the experiment defines c_0 as an encryption of m_b or m'_b respectively. Thus, A_{ind} breaks the polynomial indistinguishability of $\mathcal{CS}^0$ if $|\Pr[T_b^{(1)} = 1] - \Pr[T_b^{(2)} = 1]|$ is non-negligible.

Conclusion of Proof of Proposition. From Property (2a) of Definition 10 we conclude that the pairs $(c_0, \mathsf{Mall}(pk_0, pk_1, c_0, m'_b, t_1))$ and $(c_0, \mathsf{Enc}^1_{pk_1}(m'_b, r))$ are identically distributed, which implies that $T_0^{(2)}$ and $T_1^{(2)}$ are identically distributed. The claims above now imply that the proposition holds. □