Simplified Submission of Inputs to Protocols

Douglas Wikström*

CSC KTH Stockholm, Sweden
dog@csc.kth.se

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 homomorphic cryptosystem. This can clearly be achieved using general techniques, but we would like a solution which is: (i) provably secure under standard assumptions, (ii) non-interactive for submittors (iii) 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.

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 [16] 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, i.e., the ciphertexts should be non-malleable during the submission phase and then become homomorphic when the mixing starts.

Formally, the proof of security fails if a semantically secure cryptosystem is used directly. When using the simulation paradigm, e.g., universally composable security [4], a mix-net is said to be secure if for every adversary attacking the mix-net

* Work partly done while at ETH Zürich, Department of Computer Science.

R. Ostrovsky, R. De Prisco, and I. Visconti (Eds.): SCN 2008, LNCS 5229, pp. 293–308, 2008.

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 zero 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.

General Case. More generally, consider a cryptographic protocol that starts with a submission phase where many parties submit ciphertexts formed with a public key pk, and a smaller group of servers hold 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 semantically secure 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 semantically secure cryptosystem.

Submission Problem. The submission problem is how to find a submission scheme such that: (I) the inputs of corrupted parties can be extracted by the simulator without using the secret key, and (II) its output is a list of ciphertexts of 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.

(i) The solution should be provably secure under standard assumptions in the plain model, i.e., without any random oracles or generic groups.
(ii) The submission of inputs should be non-interactive for submittors.
(iii) The solution should be very efficient for all parties in terms of computation and communication. More precisely, when N and k denotes the number of submittors and servers respectively, then the computational complexity of each submitter should be independent of k, the communication complexity of each server should be independent of N, and the computational complexity of each server should be of the form $T(k)+T'(N)$ for some functions T and T'.

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 semantically secure cryptosystem. We classify the solutions in the literature and some extensions as follows:

1. A non-interactive proof of knowledge in the *random oracle model* is used, either using the Naor and Yung double ciphertext trick, or with rewinding. Such solutions are typically very efficient, but unfortunately only heuristically secure [5]. Note that the CCA2-secure cryptosystems in the random oracle model given by Shoup and Gennaro [29] may be viewed as instantiations of this solution.
2. An *interactive* proof of knowledge [17] is used, either with a straight-line extractor in the public key setting using the Naor and Yung [20] double-ciphertext trick, or with a rewinding extractor.
3. A non-interactive proof of knowledge using *general techniques* [2] is used. This is *not efficient* for either the submittor or the servers, even using the recent techniques of Groth et al. [18]. One could also use the fairly general zero-knowledge proofs based on homomorphic encryption of Damgård, Fazio and Nicolosi [12], but this requires non-standard assumptions. Note that using a non-interactive proof in this way is essentially the construction of a CCA2-secure cryptosystem under general assumptions given by Sahai [27] based on the Naor-Yung double-ciphertext trick, but starting from a concrete semantically secure cryptosystem with useful structure.
4. 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, and the servers must interact for each submitted ciphertext to verify its proof.
5. Non-interactive secret-key proofs using Cramer and Damgård [7] could be used. Their technique allows a prover and verifier to set up a secret key in a preliminary phase that later allows the prover to show that it behaves in a way consistent with the secret keys. Their scheme could be used in two ways. Either each submittor would take part in a protocol in the preliminary phase where the needed correlated secret keys are generated, or the servers would generate secret keys relative each other that allow them to prove that they performed the verification of a Cramer-Shoup ciphertext correctly. In the former case, interaction is moved to a preliminary phase, but each submittor must still *interact* with the servers and the servers must store a secret key for each submittor. In the latter case, submitting is non-interactive for the submittor, but each server must send and receive a non-interactive proof for each sender, i.e., its *communication complexity* with the other servers is *linear* in N.

6. An arbitrary CCA2-secure cryptosystem is used and ciphertexts are translated into suitable semantically secure ciphertexts using *general multiparty computation* techniques. This is inefficient both in terms of computation and communication.
7. A special CCA2-secure cryptosystem such that a ciphertext can be transformed more easily into a new ciphertext for the basic semantically secure scheme needed by the servers is used. We list the solutions of this type we are aware of below.
   (a) Canetti and Goldwasser [6] and Lysyanskaya and Peikert [19] have given CCA2-secure cryptosystems with distributed decryption which allows transforming ciphertexts into ciphertexts for semantically secure 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 semantically secure 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* and requires communication linear in the number of ciphertexts between the servers to verify validity.
8. Prabhakaran and Rosulek [25] present a re-randomizable and replayable CCA secure cryptosystem where one could view the transformation as trivial, i.e., nothing would be done with the ciphertexts before handing them to the underlying protocol.
   This work is interesting, but we are not aware of any (interesting) underlying protocol that accepts input ciphertexts of their cryptosystem. In fact, the authors point out that it can *not* be used directly to construct a mix-net, and even if that would be possible it would give an *inefficient* mix-net due to the larger and more complex ciphertexts.
   We remark that our work was publicly available [32] before their work was published.

To summarize, there are numerous solutions to the submission problem which satisfies properties (I) and (II), but no such solution has the properties (i)-(iii) listed above for any interesting underlying protocol.

What Is Used In Existing Mix-Nets? There are numerous proposed mix-nets with informal security arguments. If the submission problem is considered at all, the Fiat-Shamir heuristic is used (mostly even without a proof in the random oracle model). In the provably secure mix-nets either a secret sharing based solution is used [30], or an interactive proof of knowledge is used [31,33].

1.2 Our Contribution

We give a simple solution to the submission problem that is efficient both in terms of computation and communication. Although the solution is nothing

more than an observation on the Cramer-Shoup cryptosystem, it is novel and important, since it gives a truly *practical and provably secure* way for senders to submit their inputs to a mix-net, and this solution has eluded researchers ever since the Cramer-Shoup cryptosystem appeared 10 years ago.

The Idea. Recall the original Cramer and Shoup scheme [10]. The cryptosystem is deployed in a group G_q of prime order q 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 chooses $x_0, x_1, y_0, y_1, z \in \mathbb{Z}_q$ randomly 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 generates a collision-free hash function H. Finally, it outputs $(pk, sk) = ((H, g_0, g_1, c, d, h), (x_0, x_1, y_0, y_1, z))$. 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 $(u_0, u_1, e, v) = (g_0^r, g_1^r, h^r m, c^r d^{rH(u_0,u_1,e)})$. To decrypt a tuple $(u_0, u_1, e, v) \in G_q^4$ using the secret key sk the decryption algorithm tests if $u_0^{x_0} u_1^{x_1} (u_0^{y_0} u_1^{y_1})^{H(u_0,u_1,e)} = v$ to check the validity of the ciphertext. If so it outputs e/u_0^z, and otherwise the unit element of the group.[1]

Note that $h = g^z$ and z have the form of an El Gamal [16] 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.

Generalizing and Applying the Idea. To allow us to generalize the observation about the original Cramer-Shoup scheme and identify a class of cryptosystems for which it applies, we introduce the notion of an *augmented cryptosystem* which contains another cryptosystem as a component. In applications, the latter cryptosystem will have some useful structure, e.g., be homomorphic, that allows more efficient and simpler protocols. We also introduce a strengthened variation of CCA2-security called *submission security* and observe that the generic scheme of Cramer and Shoup [11] already satisfies this stronger definition. In the full version [32] we also illustrate the use of the new notion by applying it to general secure function evaluation, which strictly generalizes the notion of a mix-net.

The real efficiency gain from using our technique obviously depends on the application, but it is clear that when the number of submittors N is large the complexity of our solution based on the El Gamal cryptosystem is close to that of the most efficient heuristic solution in the random oracle model. Due to the cost of evaluating a pairing we also expect it to out-perform any solution based on elliptic curves with pairings.

Limitations of Our Approach. When using our solution, no new inputs can be accepted after part of the secret key is revealed. This is a minor drawback in the targeted applications, since we have a large number of submitting parties and executions of the underlying protocol are infrequent. When a new session

[1] In [10] a special symbol $\perp$ is output if the test fails, but this is only to simplify the analysis. Any fixed output works just as well.

is executed the servers simply generate a new key. However, it may be useful to re-use the public key of the basic cryptosystem in the underlying protocol. Thus, our definitions require that the augmentation can be replaced by a freshly generated augmentation without any loss in security. This allows using several independent augmentations 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. [6], our approach is not reasonable, since users requesting a decryption expect the result immediately.

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 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 *augmented* cryptosystems that given part of the secret key allow conversion of a ciphertext into a ciphertext of another *basic* cryptosystem. In applications, the basic 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 a cryptosystem* $\mathcal{CS}^B = (\mathsf{Kg}^B, \mathsf{Enc}^B, \mathsf{Dec}^B)$ *if there exists an* augmentation algorithm $\mathsf{Aug} \in \mathrm{PPT}$ *and a* stripping algorithm $\mathsf{Strip} \in \mathrm{PT}$ *such that:*

1. *On input* 1^n, Kg *computes* $(pk^B, sk^B) = \mathsf{Kg}^B(1^n)$ *and* $(pk^A, sk^A) = \mathsf{Aug}(pk^B)$, *and outputs* $(pk, sk) = ((pk^A : pk^B), (sk^A : sk^B))$.
2. *On input* $((sk^A : sk^B), c)$, Dec *outputs* $\mathsf{Dec}^B_{sk^B}(\mathsf{Strip}_{pk,sk^A}(c))$.

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, but we are interested in non-trivial augmentations where $\mathcal{CS}^B$ has structural properties useful in the construction of protocols.

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^A. 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 on which particular properties of the basic cryptosystem that should be considered useful.

2.1 Submission Security of Augmented Cryptosystems

Recall the game considered in the definition of CCA2-security [20,13,26]. 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 is parameterized by a bit b and returns a challenge ciphertext of 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 difference in distribution of b' when $b = 0$ or $b = 1$ respectively, should be negligible. Consider the following game.

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

$$
\begin{aligned}
(pk^B, sk^B) &\leftarrow \mathsf{Kg}^B(1^n) && \textit{// Basic keys}\\
(pk^A_j, sk^A_j) &\leftarrow \mathsf{Aug}(pk^B) \quad \textit{for } j = 1, 2, 3, \ldots && \textit{// Augmentations}\\
(pk_j, sk_j) &\leftarrow \big((pk^A_j : pk^B), (sk^A_j : sk^B)\big) && \textit{// Augmented keys}\\
(i, m_0, m_1, state) &\leftarrow A^{pk^A_{(\cdot)}, sk^A_{(\cdot)}, \mathsf{Dec}_{sk_{(\cdot)}}(\cdot)}(\textit{choose}, pk^B) && \textit{// Choice of challenges}\\
c &\leftarrow \mathsf{Enc}_{pk_i}(m_b) && \textit{// Challenge ciphertext}\\
d &\leftarrow A^{pk^A_{(\cdot)}, sk^A_{(\cdot)}, \mathsf{Dec}_{sk_{(\cdot)}}(\cdot)}(\textit{guess}, state) && \textit{// Guess of adversary}
\end{aligned}
$$

The experiment returns 0 if $\mathsf{Dec}_{sk_{(\cdot)}}(\cdot)$ was queried on (i, c) or if it was queried on (j, c') for some c' after $sk^A_{(\cdot)}$ was queried on j. Otherwise the experiment returns d.

In the game above, the adversary is given a public key pk^B of the basic cryptosystem. It can request that the experiment generates an augmentation $(pk^A_j, sk^A_j) = \mathsf{Aug}(pk^B)$, stores $(pk_j, sk_j) = ((pk^A_j : pk^B), (sk^A_j : sk^B))$, and returns $pk_j = (pk^A_j : pk^B)$ to the adversary. This is done by submitting the integer j to its $pk^A_{(\cdot)}$-oracle. Any subsequent identical queries j give the same pk_j. It can ask decryption queries. This is done by submitting an index and ciphertext pair (j, c') to its $\mathsf{Dec}_{sk_{(\cdot)}}(\cdot)$-oracle. It can request that the experiment reveals an augmentation sk^A_j by submitting j to its $sk^A_{(\cdot)}$-oracle, but after such a query no more decryption queries of the form (j, c') for some ciphertext c' are allowed. Then the adversary must choose an index i and two challenge messages m_0 and m_1. The game is parameterized by a bit b and returns a challenge ciphertext of the form $c = \mathsf{Enc}_{pk_i}(m_b)$. The adversary is then again allowed to: ask for more fresh public keys, ask more decryption queries with the exception of decryption of (i, c), and request more augmentations or augmentation keys. Finally, it outputs a guess b' of b. If the cryptosystem is submission secure, then the difference in distributions of b' with $b = 0$ or $b = 1$ respectively should be negligible.

We could equivalently have defined a game where the game only generates an augmentation if requested to do so by the adversary, but the above is conceptually simpler.

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

Example 1. A simple example of a submission secure cryptosystem can be derived from the scheme of Sahai [27] based on the Naor and Yung double ciphertext trick [20]. A semantically secure cryptosystem $\mathcal{CS}^B$ is given and a CCA2-secure cryptosystem $\mathcal{CS} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ is constructed as follows. To generate a public key, compute $(pk_0^B, sk_0^B) = \mathsf{Kg}^B(1^n)$ and $(pk_1^B, sk_1^B) = \mathsf{Kg}^B(1^n)$, and generate a common reference string CRS. Then output the key pair $(pk, sk) = ((pk_0^B : pk_1^B, CRS), (sk_0^B : sk_1^B))$. To encrypt a message m, output $(c_0, c_1, \pi) = (\mathsf{Enc}^B_{pk_0^B}(m), \mathsf{Enc}^B_{pk_1^B}(m), \pi)$, where π is a simulation sound non-interactive adaptively zero-knowledge proof (NIZKP) that the same message is encrypted in c_0 and c_1. To decrypt, verify the NIZKP and output $\mathsf{Dec}^B_{sk_0}(c_0)$ or 0 depending on if the NIZKP was accepted or not. The augmentation algorithm Aug takes (pk_1^B, CRS) as input and outputs $(pk_0^B, sk_0^B) = \mathsf{Kg}^B(1^n)$. The stripping algorithm Strip checks the NIZKP and outputs c_0 or $\mathsf{Enc}_{pk_0^B}(0, 0)$ depending on if the NIZKP was accepted or not.

3 Generic Cramer-Shoup Is Submission Secure

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 semantically secure cryptosystem is quite easy to see from their security proof. On the other hand we need to *prove* that this is indeed the case. Thus, we recall their scheme and prove this fact, 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).

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 $\Lambda[X, L, W, R] \in [I_n]$ (the support of 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 every $A \in \mathrm{PT}^*$: $|\Pr[A(\Lambda, x) = 1] - \Pr[A(\Lambda, x') = 1]|$ 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 every $s \in S$, $x, x' \in X$ with $x \notin L \cup \{x'\}$, and $\pi_x, \pi'_x, \pi, \pi' \in \Pi$, $\Pr_k[H_k(x, \pi_x) = \pi \wedge H_k(x', \pi'_x) = \pi' \mid \alpha(k) = s]$ 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.*

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

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 $i \leq i_l$ 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$_2$.*

Proof. 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.

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 $\Lambda \in [I_n]$ (the support of 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 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 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 $(pk, sk) = ((\widehat{s} : \Lambda, s), (\widehat{k} : k))$.

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 0.

We have not modified the cryptosystem, except that we have introduced a colon in the notation to distinguish the two parts of the public and secret keys as needed in the definition of submission security, and we have replaced the special symbol $\perp$ by the zero plaintext.

Write $\mathcal{CS} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$, and let $\mathcal{CS}^B = (\mathsf{Kg}^B, \mathsf{Enc}^B, \mathsf{Dec}^B)$ be the underlying basic cryptosystem defined as follows. It has public key (Λ, s) and secret key k. 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 $e - \mathsf{PEval}_0(\Lambda, k, x)$. It is clear that $\mathcal{CS}$ is an augmentation of $\mathcal{CS}^B$, since we can define $\mathsf{Aug}(\Lambda, s) = \mathsf{Gen}_1(1^n, \Lambda)$ and define $\mathsf{Strip}_{pk,\widehat{k}}(x, e, \widehat{\pi})$ to output (x, e) if $\mathsf{PEval}_1(\Lambda, \widehat{k}, x, e) = \widehat{\pi}$ and otherwise $\mathsf{Enc}_{pk^B}(0, 0)$.

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

Proposition 1. *If $\mathbb{M}$ is a hard subset membership problem, then $\mathcal{CS}$ is a submission secure augmentation of $\mathcal{CS}^B$.*

The key observations needed to extend Cramer's and Shoup's proof of CCA2-security are:

1. The projective property of hash proofs implies that proofs computed using a witness and hash proofs computed using the private key $\widehat{k}$ are *identical* (indeed this is how a hash proof is verified). This means that the holder of $\widehat{k}$ can "perfectly simulate" proofs without the witness, i.e., *even if $\widehat{k}$ is made public* the "simulated proof" looks *exactly* like a proof computed using a witness.
2. The soundness of proofs computed by an adversary *before* $\widehat{k}$ is made public, is not decreased by the publishing of $\widehat{k}$.

The generic Cramer-Shoup scheme generalizes several concrete schemes described in [11], such as the El Gamal based scheme in the introduction, but also schemes based on the Paillier cryposystem [22]. Both schemes are common in efficient protocols.

3.3 Proof of 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 $\mathsf{Exp}^{\mathsf{sub}-b}_{\mathcal{CS},\mathcal{CS}^B,A}(n)$ 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 $\widehat{\pi} = \mathsf{PEval}_1(\Lambda, \widehat{k}_i, x, e) = \widehat{H}_{\widehat{k}_i}(x, e)$, where i is the challenge index chosen by the adversary. 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 L or $X \setminus L$. 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 $x \in X \setminus L$. From the hardness of $\mathbb{M}$ follows that $|\Pr[T_b^{(0)} = 1] - \Pr[T_b^{(1)} = 1]|$ is negligible. □

Denote by $(i_j, (\pi_j, e_j, \widehat{\pi}_j))$ the jth query to the decryption oracle $\mathsf{Dec}_{sk_{(\cdot)}}(\cdot)$, and let j_l be the index of the last query before the adversary outputs its choice of challenge index and messages. Denote by $(x, e, \widehat{\pi})$ the challenge ciphertext, and let E be the event that A asks a decryption query $(i_j, (\pi_j, e_j, \widehat{\pi}_j))$ with $\widehat{H}_{\widehat{k}_{i_j}}(x_j, e_j) = \widehat{\pi}_j$, $x_j \in X \setminus L$, and $j \leq j_l$ or $x_j \neq x$, for some index j before it requests $sk^A_{i_j}$ from its $sk^A_{(\cdot)}$-oracle (it may of course not ever do this).

Claim 2. $\Pr[E]$ is negligible.

Proof. Let Q be the total number of queries to the augmentation oracle $pk_{(\cdot)}$ made by the adversary. Without loss we assume that the adversary asks the queries $l = 1, \ldots, Q$.

Define A^l_{uni} for $l = 1, \ldots, Q$ to be the machine that simulates $T_b^{(1)}$ and takes part in Experiment 2. The simulation is modified in that: $\widehat{s}_l$ is defined as the hash proof key received in Experiment 2, whenever $T_b^{(1)}$ needs to check a hash proof as $\mathsf{PEval}_1(\Lambda, \widehat{k}_l, x_j, e_j) = \widehat{H}_{\widehat{k}_l}(x_j, e_j)$ it simply queries its $\tau_{\widehat{k}_l}(\cdot, \cdot)$-oracle in Experiment 2 with (x_j, e_j) instead, and when it needs to compute $\mathsf{PEval}_1(\Lambda, \widehat{k}_l, x, e) = \widehat{H}_{\widehat{k}_l}(x, e) = \widehat{\pi}$ it outputs (x, e) and waits for $\widehat{H}_{\widehat{k}_l}(x, e) = \widehat{\pi}$ from the experiment instead. The computational universal$_2$ property and the union bound then implies the claim.

Note that the computational universal$_2$ property can be applied despite that the experiment reveals private hash proof keys, since by definition of submission security the adversary only wins if it never asks a decryption query after this point. 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 in nature.

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 since Π is a group. The claims above now imply that $|\Pr[T_0^{(0)} = 1] - \Pr[T_1^{(0)} = 1]|$ is negligible.

4 Applications of Submission Security

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 [28]. In the full version [32] we also illustrate how the notion of submission secure augmented cryptosystems can be used to construct and analyze the submission phase of a protocol in a modularized way for general secure function evaluation, and explain how this generalizes the mix-net setting in the informal description.

4.1 Informal Description of Application to a Mix-Net

There are many senders $S_1, \ldots, S_N$ and few mix-servers $M_1, \ldots, M_k$, e.g., $N = 10^4$ and $k = 10$. In a joint key generation phase the mix-servers generate a joint public key (g, h) such that each mix-server holds a verifiable secret share s_j of the joint secret key z such that $h = g^z$. This can be done using Feldman verifiable secret sharing [14]. To submit a message $m_i \in G_q$, a sender S_i 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 $\{(u_{j,i}, e_{j,i})\} = \{(u_{j-1,\pi_j(i)} g^{t_{j,i}}, e_{j-1,\pi_j(i)} h^{t_{j,i}})\}$, where $t_{j,i} \in \mathbb{Z}_q$ and π_j are 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 using their shares s_j, 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 correctness, each mix-server also proves in zero-knowledge that it processed the list of ciphertexts correctly. This is done using a so called proof of a shuffle [21,15].

Unfortunately, the construction is completely insecure [24], 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_i, m_i^a) in the final output, where m_i is the message sent by S_i. This reveals the message m_i 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 [30,31,33]) 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 set $g_0 = g$ and let the mix-servers generate $g_1 \in G_q$, $x_0, x_1, y_0, y_1 \in \mathbb{Z}_q$, $c = g_0^{x_0} g_1^{x_1}$, and $d = g_0^{y_0} g_1^{y_1}$, where x_0, x_1, y_0, y_1 are verifiably secret shared among the mix-servers. This can be done using Pedersen verifiable secret sharing [23] and takes place after the joint key h is generated. This gives a Cramer-Shoup key pair $((H, g_1, c, d : g_0, h), (x_0, x_1, y_0, y_1 : z))$ with verifiably secret shared secret key. 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 starting 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-servers process the El Gamal ciphertexts as explained above.

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.

Acknowledgments

I thank Eike Kiltz for helpful discussions, and I thank Ronald Cramer for answering my questions about the relation between the generic Cramer-Shoup scheme and its concrete instantiations.

References

1. Abe, M., Cramer, R., Fehr, S.: Non-interactive distributed-verifier proofs and proving relations among commitments. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 206–223. Springer, Heidelberg (2002)
2. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. In: 20th ACM Symposium on the Theory of Computing (STOC), pp. 103–118. ACM Press, New York (1988)
3. Boneh, D., Boyen, X., Halevi, S.: Chosen ciphertext secure public key threshold encryption without random oracles. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 226–243. Springer, Heidelberg (2006)
4. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 136–145. IEEE Computer Society Press, Los Alamitos (2001), `http://eprint.iacr.org`
5. Canetti, R., Goldreich, O., Halevi, S.: The random oracle model revisited. In: 30th ACM Symposium on the Theory of Computing (STOC), pp. 209–218. ACM Press, New York (1998)
6. Canetti, R., Goldwasser, S.: An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 90–106. Springer, Heidelberg (1999)
7. Cramer, R., Damgård, I.: Secret-key zero-knowlegde and non-interactive verifiable exponentiation. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 223–237. Springer, Heidelberg (2004)
8. Cramer, R., Damgård, I., Ishai, Y.: Share conversion, pseudorandom secret-sharing and applications to secure computation. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 342–362. Springer, Heidelberg (2005)
9. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
10. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
11. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption (June 1999), `http://homepages.cwi.nl/~cramer/`
12. Damgård, I., Fazio, N., Nicolosi, A.: Non-interactive zero-knowledge from homomorphic encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 41–59. Springer, Heidelberg (2006)
13. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. In: 23rd ACM Symposium on the Theory of Computing (STOC), pp. 542–552. ACM Press, New York (1991)
14. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: 28th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 427–438. IEEE Computer Society Press, Los Alamitos (1987)
15. Furukawa, J., Sako, K.: An efficient scheme for proving a shuffle. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 368–387. Springer, Heidelberg (2001)
16. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)

17. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing 18(1), 186–208 (1989)
18. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for np. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006)
19. Lysyanskaya, A., Peikert, C.: Adaptive security in the threshold setting: From cryptosystems to signature schemes. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 331–350. Springer, Heidelberg (2001)
20. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM Symposium on the Theory of Computing (STOC), pp. 427–437. ACM Press, New York (1990)
21. Neff, A.: A verifiable secret shuffle and its application to e-voting. In: 8th ACM Conference on Computer and Communications Security (CCS), pp. 116–125. ACM Press, New York (2001)
22. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
23. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)
24. Pfitzmann, B., Pfitzmann, A.: How to break the direct RSA-implementation of mixes. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 373–381. Springer, Heidelberg (1990)
25. Prabhakaran, M., Rosulek, M.: Rerandomizable rcca encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 517–534. Springer, Heidelberg (2007)
26. Rackoff, C., Simon, D.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)
27. Sahai, A.: Non-malleable non-interactive zero-knowledge and adaptive chosen-ciphertext security. In: 40th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 543–553. IEEE Computer Society Press, Los Alamitos (1999)
28. Sako, K., Killian, J.: Reciept-free mix-type voting scheme. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995)
29. Shoup, V., Gennaro, R.: Securing threshold cryptosystems against chosen ciphertext attack. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 1–16. Springer, Heidelberg (1998)
30. Wikström, D.: A universally composable mix-net. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 315–335. Springer, Heidelberg (2004)
31. Wikström, D.: A sender verifiable mix-net and a new proof of a shuffle. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 273–292. Springer, Heidelberg (2005)
32. Wikström, D.: Simplified submission of inputs to protocols. Cryptology ePrint Archive, Report 2006/259 (2006), http://eprint.iacr.org/
33. Wikström, D., Groth, J.: An adaptively secure mix-net without erasures. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 276–287. Springer, Heidelberg (2006)