A Universally Composable Mix-Net

Douglas Wikström[1,2]

[1] Royal Institute of Technology (KTH)
KTH, Nada, S-100 44 Stockholm, Sweden
[2] Swedish Institute of Computer Science (SICS)
Box 1263, S-164 29 Kista, Sweden
douglas@sics.se

Abstract. A mix-net is a cryptographic protocol executed by a set of mix-servers that provides anonymity for a group of senders. The main application is electronic voting.
Numerous mix-net constructions and stand-alone definitions of security are proposed in the literature, but only partial proofs of security are given for most constructions and no construction has been proved secure with regards to any kind of composition.
We define an ideal mix-net in the universally composable security framework of Canetti [6]. Then we describe a mix-net based on Feldman [13] and using similar ideas as Desmedt and Kurosawa [10], and prove that it securely realizes the ideal mix-net with respect to static adversaries that corrupt a minority of the mix-servers and arbitrarily many senders.
The mix-net executes in a hybrid model with access to ideal distributed key generation, but apart from that our only assumption is the existence of a group in which the Decision Diffie-Hellman Problem is hard.
If there are relatively few mix-servers or a strong majority of honest mix-servers our construction is practical.

1 Introduction

The notion of a mix-net was invented by Chaum [7]. Properly constructed a mix-net takes a list of cryptotexts and outputs the cleartexts permuted using a secret random permutation. Usually a mix-net is realized by a set of mix-servers organized in a chain that collectively execute a protocol. Each mix-server receives a list of encrypted messages from the previous mix-server, transforms them, using partial decryption or random re-encryption, reorders them, and outputs the result. The secret permutation is shared by the mix-servers.

1.1 Previous Work

Chaum's original "anonymous channel" [7, 36] enables a sender to send mail anonymously. When constructing election schemes [7, 14, 37, 43, 35] a mix-net can be used to ensure that the vote of a given voter can not be revealed. Abe gives an efficient construction of a general mix-net [1], and argues about its properties. Jakobsson has written (partly with Juels) more general papers on the topic of mixing [24–26] focusing on efficiency. Furukawa and Sako [15], and Neff [33] respectively have recently found efficient proofs of a correct shuffle, but these proposals have incomplete or flawed analysis. Groth [23] builds on Neff's ideas to form an abstract protocol for any homomorphic cryptosystem.

Desmedt and Kurosawa [10] describe an attack on a protocol by Jakobsson [24]. Similarly Mitomo and Kurosawa [32] exhibit a weakness in another protocol by Jakobsson [25]. Pfitzmann has given some general attacks on mix-nets [40, 39], and Michels and Horster give additional attacks in [31]. Wikström [46] gives several attacks for a protocol by Golle et al. [22]. He also gives attacks for the protocols by Jakobsson [25] and Jakobsson and Juels [27]. Abe [2] has independently found related attacks.

Canetti [6] and independently Pfitzmann and Waidner [41], proposed security frameworks for reactive processes. We use the former framework. Both frameworks has composition theorems, and are based on older definitional work. The initial ideal-model based definitional approach for secure function evaluation is informally proposed by Goldreich, Micali, and Wigderson in [18]. The first formalizations appear in Goldwasser and Levin [19], Micali and Rogaway [30], and Beaver [3]. Canetti [5] presents the first definition of security that is preserved under composition. See [5, 6] for an excellent background on these definitions.

1.2 Contribution

The large number of attacks and flawed analysis for mix-net constructions, e.g. [10, 32, 40, 39, 31, 46, 2] and [33, 15] respectively, suggest that constructing a secure mix-net is hard. Previous work on mix-nets gives ad-hoc definitions of security, and most provide proofs in heuristic models. We take a broader view and present the first mix-net *provably secure* in the UC-security framework. To achieve this we introduce a natural definition of a UC-secure mix-net, and avoid all two-party proofs of knowledge. Instead we introduce multi-verifier proofs of knowledge that exploit the potential of an honest majority of mix-servers.

1.3 Outline of the Paper

The paper is organized as follows. First we define ideal functionalities corresponding to the notions of a mix-net, a bulletin board, distributed key generation, and multi-verifier zero-knowledge. Then we describe a generic mix-net running in a hybrid model with access to these ideal functionalities (except the ideal mix-net) that securely realizes the ideal mix-net. This is followed by protocols that securely realize a proof of knowledge of a cleartext of an El Gamal cryptotext, and a proof of knowledge of the correctness of a decrypt-shuffle respectively. Finally we use the composition theorem of Canetti [6] to compose our protocols with each other and with the universally composable authenticated broadcast presented by Goldwasser and Lindell [20]. This gives a universally composable mix-net in a hybrid model with ideal distributed key generation. In this conference version we only give shortened proofs of Lemma 2 and Lemma 4. The full version of this paper [47] provides proofs of all claims.

1.4 Notation

Throughout $S_1, \ldots, S_N$ will denote senders and $M_1, \ldots, M_k$ mix-servers. All participants are modeled as interactive Turing machines. We abuse notation and use P_i and M_j to denote both the machines themselves and their identity. We denote the set of permutations of N elements by Σ_N. We use the term "randomly" instead of "uniformly and independently at random". We assume that G_q is a group of prime order q with generator g for which the Decision Diffie-Hellman Assumption holds. Informally the assumption says that it is hard to distinguish the distributions $(g^\alpha, g^\beta, g^{\alpha\beta})$ and $(g^\alpha, g^\beta, g^\gamma)$ when $\alpha, \beta, \gamma \in \mathbb{Z}_q$ are randomly chosen.

We review the El Gamal [12] cryptosystem employed in G_q. The private key x is generated by choosing $x \in \mathbb{Z}_q$ randomly. The corresponding public key is $y = g^x$. Encryption of a message $m \in G_q$ using the public key y is given by $E_y(m, r) = (g^r, y^r m)$, where r is chosen randomly from $\mathbb{Z}_q$, and decryption of a cryptotext on the form $(u, v) = (g^r, y^r m)$ using the private key x is given by $D_x(u, v) = u^{-x} v = m$. Tsionis and Yung [45] shows that the El Gamal cryptosystem is semantically secure [21, 29] under the DDH-assumption.

1.5 The UC-Security Framework

In this conference version we only give a short informal review of the UC-framework. For details we refer the reader to Canetti [6] or the full version of this paper [47].

The core of the framework consists of the real model, the ideal model, and many different hybrid models. In all models the corresponding adversary may corrupt a certain fraction of the parties.

The real model is a model of real world computing, i.e. a list of interactive Turing machines execute a protocol over an asynchronous authenticated open network. The real adversary can see all communication and decide when messages are delivered. The ideal model contains an ideal functionality, i.e. a trusted party, that defines a service we wish to implement. Thus a protocol in the ideal model is trivial and consists of machines that forwards any input to the ideal functionality, and gives any output from the ideal functionality as output. The ideal adversary decides when messages are delivered from the ideal functionality but it can not see any contents. An ideal functionality is considered secure by definition. To be able to seamlessly move from a real model to an ideal model there are many hybrid models. A protocol running in a hybrid model is a list of interactive Turing machines that has access to some set of ideal functionalities.

The definition of security is based on the simulation paradigm. A protocol is said to securely realize an ideal functionality if for any real adversary in the real model, there is an ideal adversary in the ideal model that has the same advantage. In contrast to classical definitions the distinguisher is present during the execution and may influence the adversary based on part of the transcript.

The definition of security allows secure composition of protocols, i.e. given a protocol secure in a hybrid model, and protocols that securely realize all ideal functionalities in use, it is trivial to construct a secure protocol in the real model.

The notion of a communication model, $\mathcal{C}_{\mathcal{I}}$, used below is not explicit in Canetti [6]. It works as a router between participants and between participants and ideal functionalities. Given input $(A, B, C, \ldots)$ it interprets A as the receiver of $(B, C, \ldots)$. The adversary can not read the correspondence with ideal functionalities, but it has full control over when a message is delivered.

Throughout we consider the adversary model below, and we explicitly say when a result holds only with regards to blocking adversaries.

Definition 1. *We define $\mathcal{M}_{B(k)}$ to be the set of* static adversaries *that corrupt less than $B(k)$ participants of the mix-server type M_j, and arbitrarily many participants of the sender type P_i.*

2 Ideal Functionalities

No definition of an ideal mix-net in the UC-security framework has been given in the literature. Below we give a natural definition corresponding to a mix-net that outputs the cleartexts. The term mix-net is sometimes used also for constructions that do not decrypt the inputs, but we do not consider this here. We assume that each sender only sends one message.

Throughout we implicitly assume that a message handed to an ideal functionality that is not on the forms prescribed in its definition is returned to the sender immediately. In particular this includes verifying membership in G_q when appropriate. We use the same convention for definitions of protocols.

Functionality 1 (Mix-Net). The ideal functionality for a *mix-net*, $\mathcal{F}_{\mathrm{MN}}$, running with mix-servers $M_1, \ldots, M_k$, senders $P_1, \ldots, P_N$, and ideal adversary $\mathcal{S}$ proceeds as follows

1. Initialize a list $L = \emptyset$, and set $J_P = \emptyset$ and $J_M = \emptyset$.
2. Suppose $(P_i, \texttt{Send}, m_i)$, $m_i \in G_q$, is received from $\mathcal{C}_{\mathcal{I}}$. If $i \notin J_P$, set $J_P \leftarrow J_P \cup \{i\}$, and append m_i to the list L. Then hand $(\mathcal{S}, P_i, \texttt{Send})$ to $\mathcal{C}_{\mathcal{I}}$.
3. Suppose $(M_j, \texttt{Run})$ is received from $\mathcal{C}_{\mathcal{I}}$. Set $J_M \leftarrow J_M \cup \{j\}$. If $|J_M| \geq k/2$, then sort the list L lexicographically to form a list L', and hand $((\mathcal{S}, M_j, \texttt{Output}, L'), \{(M_l, \texttt{Output}, L')\}_{l=1}^{k})$ to $\mathcal{C}_{\mathcal{I}}$. Otherwise, hand $\mathcal{C}_{\mathcal{I}}$ the list $(\mathcal{S}, M_j, \texttt{Run})$.

Most constructions given in the literature assume the existence of an authenticated bulletin board, but this assumption is rarely formalized.

Functionality 2 (Bulletin Board). The ideal *bulletin board* functionality, $\mathcal{F}_{\mathrm{BB}}$, running with participants $P_1, \ldots, P_k$ and ideal adversary $\mathcal{S}$.

1. $\mathcal{F}_{\mathrm{BB}}$ holds a database indexed on integers. Initialize a counter $c = 0$.
2. Upon receiving $(P_i, \texttt{Write}, m_i)$, $m_i \in \{0,1\}^*$, from $\mathcal{C}_{\mathcal{I}}$, store (P_i, m_i) under the index c in the database, hand $(\mathcal{S}, \texttt{Write}, c, P_i, m_i)$ to $\mathcal{C}_{\mathcal{I}}$, and set $c \leftarrow c+1$.
3. Upon receiving $(P_j, \texttt{Read}, c)$ from $\mathcal{C}_{\mathcal{I}}$ check if a tuple (P_i, m_i) is stored in the database under c. If so hand $((\mathcal{S}, P_j, \texttt{Read}, c, P_i, m), (P_j, \texttt{Read}, c, P_i, m_i))$ to $\mathcal{C}_{\mathcal{I}}$. If not, hand $((\mathcal{S}, P_j, \texttt{NoRead}, c), (P_j, \texttt{NoRead}, c))$ to $\mathcal{C}_{\mathcal{I}}$.

Goldwasser and Lindell [20] show that an authenticated broadcast can be securely realized with respect to *blocking* $\mathcal{M}_{k/2}$-adversaries. On the other hand Lindell, Lysyanskaya and Rabin [28] show that composable authenticated broadcast can not be realized for *non-blocking* $\mathcal{M}_B$-adversaries if $B > k/3$. The following lemma follows from [20].

Lemma 1. *There exists a protocol* π_{BB} *that securely realizes* $\mathcal{F}_{\mathrm{BB}}$ *with respect to blocking* $\mathcal{M}_{k/2}$*-adversaries.*

The mix-servers must somehow set up distributed El Gamal keys, but we do not consider this problem here. We only note that the problem was first addressed by Pedersen [38], and that Gennaro et al. [17] discovered a flaw in his approach, and a solution to the problem. Unfortunately their protocol is given in a different model, and can not be applied here directly. Below, the joint secret key $x = \sum_{j=1}^{k} x_j$ and the corresponding public key $y = \prod_{j=1}^{k} y_j = g^x$ are implicit. Any individual participant can compute y, but not x.

Functionality 3 (Distributed El Gamal Key Generation). The ideal *Distributed El Gamal Key Generation over* G_q, $\mathcal{F}_{\mathrm{KG}}$, running with mix-servers $M_1, \ldots, M_k$, senders $P_1, \ldots, P_N$, and ideal adversary $\mathcal{S}$ proceeds as follows.

1. Initialize sets $J_j = \emptyset$ for $j = 0, \ldots, k$.
2. Until $|J_0| = k$ wait for $(M_j, \mathtt{MyKey}, x_j, y_j)$ from $\mathcal{C}_{\mathcal{I}}$ such that $x_j \in \mathbb{Z}_q$, $y_j = g^{x_j}$, and $j \notin J_0$. Set $J_0 \leftarrow J_0 \cup \{j\}$.
3. Hand $((\mathcal{S}, \mathtt{PublicKeys}, y_1, \ldots, y_k), \{(P_j, \mathtt{PublicKeys}, y_1, \ldots, y_k)\}_{j=1}^{N}, \{(M_j, \mathtt{Keys}, x_j, y_1, \ldots, y_k)\}_{j=1}^{k})$ to $\mathcal{C}_{\mathcal{I}}$.
4. If $(M_j, \mathtt{Recover}, M_l)$ is received from $\mathcal{C}_{\mathcal{I}}$, set $J_l \leftarrow J_l \cup \{j\}$. If $|J_l| \geq k/2$, then hand $((\mathcal{S}, \mathtt{Recovered}, M_l, x_l), \{(M_j, \mathtt{Recovered}, M_l, x_l)\}_{j=1}^{k})$ to $\mathcal{C}_{\mathcal{I}}$, and otherwise hand $(\mathcal{S}, M_j, \mathtt{Recover}, M_l)$ to $\mathcal{C}_{\mathcal{I}}$.

We need two different zero-knowledge proofs of knowledge. Following Canetti et al. [8] we define a single ideal zero-knowledge functionality taking a relation R as a parameter, and then give two polynomial-time recognizable relations R_{C}, and R_{DS} for the functionalities we need.

Functionality 4 (Zero-Knowledge Proof of Knowledge). Let $\mathcal{L}$ be a language given by a binary relation R. The ideal *zero-knowledge proof of knowledge* functionality of a witness w to an element $x \in \mathcal{L}$, running with provers $P_1, \ldots, P_N$, and verifiers $M_1, \ldots, M_k$, proceeds as follows.

1. Upon receipt of $(P_i, \mathtt{Prover}, x, w)$ from $\mathcal{C}_{\mathcal{I}}$, store w under the tag (P_i, x), and hand $(\mathcal{S}, P_i, \mathtt{Prover}, x, R(x, w))$ to $\mathcal{C}_{\mathcal{I}}$. Ignore further messages from P_i.
2. Upon receipt of $(M_j, \mathtt{Question}, P_i, x)$ from $\mathcal{C}_{\mathcal{I}}$, let w be the string stored under the tag (P_i, x) (the empty string if nothing is stored), and hand $((\mathcal{S}, M_j, \mathtt{Verifier}, P_i, x, R(x, w)), (M_j, \mathtt{Verifier}, P_i, R(x, w))$ to $\mathcal{C}_{\mathcal{I}}$.

The first relation corresponds to knowledge of the cleartext m, when (u, v) is interpreted as $(g^r, y^r m)$. This may be viewed as the ideal counterpart of the proof of knowledge in the heuristically non-malleable version of El Gamal described both by Tsionis and Yung [45] and Schnorr and Jakobsson [44].

Definition 2 (Knowledge of Cleartext). *Define a relation* $R_{\mathrm{C}} \subset G_q^4 \times \mathbb{Z}_q$*, by* $((g, y, u, v), r) \in R_{\mathrm{C}}$ *precisely when* $\log_g u = r$.

Although neither y nor v plays any role in the definition we keep them to emphasize the similarity with older work.

The second relation corresponds to a correct partial decryption, permutation and re-encryption of a list of El Gamal cryptotexts. This may be viewed as the ideal counterpart to the honest verifier zero-knowledge proof of knowledge presented by Furukawa et al. [16], or that of Neff [34].

Definition 3 (Knowledge of Correct Decrypt-Shuffle). *Define for each* N *a relation* $R_{\mathrm{DS}} \subset (G_q^3 \times G_q^{2N} \times G_q^{2N}) \times (\mathbb{Z}_q \times \Sigma_N \times \mathbb{Z}_q^N)$*, by*

$$((g, h, y, \{(u_i, v_i)\}_{i=1}^N, \{(u_i', v_i')\}_{i=1}^N), (x, \pi, \{r_i\}_{i=1}^N)) \in R_{\mathrm{DS}}$$

precisely when $(u_i', v_i') = (g^{r_i} u_{\pi^{-1}(i)}, h^{r_i} v_{\pi^{-1}(i)} u_{\pi^{-1}(i)}^{-x})$ *for* $i = 1, \ldots, N$*, and* $\log_g y = x$.

In an application the prover M_j holds π, r_i, and x such that $y = g^x$, and h corresponds to the remaining part of a shared key.

3 A Generic Mix-Net in a Hybrid Model

We describe a generic mix-net protocol in the $(\mathcal{F}_{\mathrm{BB}}, \mathcal{F}_{\mathrm{KG}}, \mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}, \mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}})$-hybrid model, i.e. the participants use an ideal bulletin board, ideal distributed El Gamal key generation, and ideal zero-knowledge proof systems for the relations R_{C} and R_{DS}. The structure of our mix-net corresponds closely to the mix-net implemented by Furukawa et al. [16]. Other researchers, e.g. Neff [34], have had similar ideas. Our mix-net is secure as long as a majority of the mix-servers M_j are honest. There is no bound on the number of corrupted senders P_i.

In the other common structure of a mix-net each mix-server performs a random re-encryption and permutation, and then the mix-servers jointly decrypt the output of the last mix-server. We believe that our results may be generalized to hold for such a protocol.

We abuse notation. When a message is received via a copy of the ideal communication model $\mathcal{C}_{\mathcal{I}}$, we say that it is received directly from an ideal functionality.

Informally the mix-net works as follows. Each sender encrypts its message using the El Gamal cryptosystem and proves that it knows the randomness used to do this. Then the mix-servers take turns to partially decrypt, permute, and re-encrypt the elements in the list. The output of the last mix-server is a list of permuted cleartexts.

Protocol 1 (Generic Mix-Net). The generic mix-net protocol
$\pi = (P_1, \ldots, P_N, M_1, \ldots, M_k)$ consists of senders P_i, and mix-servers M_j.

SENDER P_i. Each sender P_i proceeds as follows.

1. Wait for $(\texttt{PublicKeys}, y_1, \ldots, y_k)$ from $\mathcal{F}_{\mathrm{KG}}$, and compute $y = \prod_{\iota=1}^k y_\iota$.

2. Wait for an input $(\texttt{Send}, m_i)$, $m_i \in G_q$. Then choose $r_i \in \mathbb{Z}_q$ randomly and compute $(u_i, v_i) = E_y(m_i, r_i)$.
3. Hand $(\texttt{Prover}, (g, y, u_i, v_i), r_i)$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$.
4. Hand $(\texttt{Write}, (u_i, v_i))$ to $\mathcal{F}_{\mathrm{BB}}$.

MIX-SERVER M_j. Each mix-server M_j proceeds as follows.

1. Choose $x_j \in \mathbb{Z}_q$ randomly and hand $(\texttt{MyKey}, x_j, g^{x_j})$ to $\mathcal{F}_{\mathrm{KG}}$.
2. Wait for $(\texttt{Keys}, x_j, y_1, \ldots, y_k)$ from $\mathcal{F}_{\mathrm{KG}}$, compute $h_l = \prod_{j=l}^{k} y_j$ for $l = 1, \ldots, k$, and set $y = h_1$.
3. Wait for an input $(\texttt{Run})$, and then hand $(\texttt{Write}, \texttt{Run})$ to $\mathcal{F}_{\mathrm{BB}}$.
4. Wait until at least $k/2$ different mix-servers has written Run on $\mathcal{F}_{\mathrm{BB}}$, and let the last entry of this type be $(c_{\texttt{Run}}, M_i, \texttt{Run})$.
5. Form the list $L_* = \{(u_\gamma, v_\gamma)\}_{\gamma \in I_*}$, for some index set I_*, from the set of entries on $\mathcal{F}_{\mathrm{BB}}$ on the form $(c, P_\gamma, (u_\gamma, v_\gamma))$, where $0 \leq c < c_{\mathrm{run}}$, $\gamma \in \{1, \ldots, N\}$, and $u_\gamma, v_\gamma \in G_q$.
6. For each $\gamma \in I_*$ do the following,
   (a) Hand $(\texttt{Question}, P_\gamma, (g, y, u_\gamma, v_\gamma))$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$.
   (b) Wait for $(\texttt{Verifier}, P_\gamma, b_\gamma)$ from $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$.

   Then form $L_0 = \{(u_{0,i}, v_{0,i})\}_{i=1}^{N'}$ consisting of pairs (u_γ, v_γ) such that $b_\gamma = 1$.
7. For $l = 1, \ldots, k$ do:
   (a) If $l \neq j$, then do:
      i. Wait until an entry $(c, M_l, (\texttt{List}, L_l))$ appears on $\mathcal{F}_{\mathrm{BB}}$, where L_l is on the form $\{(u_{l,i}, v_{l,i})\}_{i=1}^{N'}$ for $u_{l,i}, v_{l,i} \in G_q$.
      ii. Hand $(\texttt{Question}, M_l, (g, h_{l+1}, y_l, L_{l-1}, L_l))$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, and wait for $(\texttt{Verifier}, M_l, b_l)$ from $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$.
      iii. If $b_l = 0$, then hand $(\texttt{Recover}, M_l)$ to $\mathcal{F}_{\mathrm{KG}}$, and wait for $(\texttt{Recovered}, M_l, x_l)$ from $\mathcal{F}_{\mathrm{KG}}$. Then define $L_l = \{(u_{l,i}, v_{l,i})\}_{i=1}^{N'} = \{D_{x_j}(u_{l-1,i}, v_{l-1,i})\}_{i=1}^{N'} = \{(u_{l-1,i}, v_{l-1,i} u_{l-1,i}^{-x_l})\}_{i=1}^{N'}$.

   (b) If $l = j$, then choose $r_{j,i} \in \mathbb{Z}_q$ and $\pi_j \in \Sigma_{N'}$ randomly, and compute
   $$L_j = \{(u_{j,i}, v_{j,i})\}_{i=1}^{N'} = \left\{ \left(g^{r_{j,i}} u_{j-1,\pi_j^{-1}(i)}, h_{j+1}^{r_{j,i}} \frac{v_{j-1,\pi_j^{-1}(i)}}{u_{j-1,\pi_j^{-1}(i)}^{x_j}} \right) \right\}_{i=1}^{N'} .$$
   Finally hand $(\texttt{Prover}, (g, h_{j+1}, y_j, L_{j-1}, L_j), (x_j, \pi_j, \{r_{j,i}\}_{i=1}^{N'}))$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, and hand $(\texttt{Write}, (\texttt{List}, L_j))$ to $\mathcal{F}_{\mathrm{BB}}$.
8. Sort $\{v_{k,i}\}_{i=1}^{N'}$ lexicographically to form a list L' and output $(\texttt{Output}, L')$.

Lemma 2. *Protocol 1 securely realizes the ideal functionality $\mathcal{F}_{\mathrm{MN}}$ in the $(\mathcal{F}_{\mathrm{BB}}, \mathcal{F}_{\mathrm{KG}}, \mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}, \mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}})$-hybrid model with respect to $\mathcal{M}_{k/2}$-adversaries under the DDH-assumption in G_q.*

Each mix-server computes $3N$ exponentiations in G_q.

Lemma 2 reduces the problem of constructing a UC-secure mix-net to the problem of constructing UC-secure realizations of the ideal functionalities $\mathcal{F}_{\mathrm{BB}}$, $\mathcal{F}_{\mathrm{KG}}$, $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$. The lemma can also viewed as an argument of the security of mix-nets where the ideal functionalities $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ are heuristically, but efficiently, realized, e.g. by zero-knowledge proofs of knowledge in the common random string model or the random oracle model (cf. [33, 15, 16, 23]).

Pfitzmann [39, 40] shows that the cryptotexts handed to a mix-net must be non-malleable [11] and a common way to ensure this is to use the cryptosystem suggested by Tsionis and Yung [45] and Schnorr and Jakobsson [44], or the Cramer-Shoup cryptosystem [9]. Both constructions are efficient and may be viewed as El Gamal augmented with a proof of knowledge, but only the latter is provably secure and both lack the extraction requirements of the UC-framework.

4 Secure Realizations of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$

We securely realize the ideal functionalities $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ in the $\mathcal{F}_{\mathrm{BB}}$-hybrid model in a reasonably practical way as long as the number of mix-servers is relatively small. A key observation is that since we are considering $\mathcal{M}_{B(k)}$-adversaries, the prover may well disclose its secret witness to all subsets consisting of at least $B(k)$ verifiers as long as it does not disclose it to any subset consisting of less than $B(k)$ verifiers.

4.1 A Realization of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ in the $\mathcal{F}_{\mathrm{BB}}$-Hybrid Model

We observe that we may view the verifiable secret sharing scheme (VSS) of Feldman [13] as a multi-verifier proof of knowledge of a logarithm, since his scheme only leaks information on the secret that is already known in our setting! Note that this protocol does *not* securely realize any natural ideal VSS functionality. The simulatability properties of the UC-framework are not satisfied.

Intuitively, the protocol works as follows. A prover shares his witness to the relation R_{C}, and uses a semantically secure cryptosystem over the authenticated bulletin board $\mathcal{F}_{\mathrm{BB}}$ to distribute the shares. The verifiers check their shares, and write the result of their verification on the bulletin board. Each verifier then checks that all verifiers accepted their shares.

Protocol 2 (Zero-Knowledge Proof of Knowledge of Cleartext). Let $t = \lceil k/2 - 1 \rceil$. The zero-knowledge proof of knowledge of a cleartext protocol $\pi = (P_1, \ldots, P_N, M_1, \ldots, M_k)$ consists of provers P_i, and verifiers M_j.

PROVER P_i.

1. Wait until $(\cdot, M_j, \mathtt{Keys}, y_{j,1}, \ldots, y_{j,N})$ appears on $\mathcal{F}_{\mathrm{BB}}$ for $j = 1, \ldots, k$.
2. Wait for input $(\mathtt{Prover}, (g, y, u_i, v_i), r_i)$, where $g, y, u_i, v_i \in G_q$ and $r_i \in \mathbb{Z}_q$.
3. Choose $a_{i,l} \in \mathbb{Z}_q$ randomly, define $p_i(x) = r + \sum_{l=1}^{t} a_{i,l} x^l$, and compute
$$\alpha_{i,l} = g^{a_{i,l}} \text{ for } l = 1, \ldots, t, \quad s_{i,j} = p_i(j) \text{ for } j = 1, \ldots, k \ , \quad \text{and}$$
$$C_{i,j} = E_{y_{j,i}}(s_{i,j}), \quad \text{for } j = 1, \ldots, k \ .$$
4. Hand $(\mathtt{Write}, \mathtt{Proof}, (g, y, u_i, v_i), (\alpha_{i,1}, \ldots, \alpha_{i,t}, C_{i,1}, \ldots, C_{i,k}))$ to $\mathcal{F}_{\mathrm{BB}}$.

VERIFIER M_j.

1. Generate El Gamal keys $(x_{j,i}, y_{j,i})$ for $i = 1, \ldots, N$, and hand $(\texttt{Write}, \texttt{Keys}, y_{j,1}, \ldots, y_{j,N})$ to $\mathcal{F}_{\mathrm{BB}}$.
2. On input $(\texttt{Question}, P_i, (g, y, u_i, v_i))$ do:
   (a) If $(\cdot, P_i, \texttt{Proof}, (g, y, u_i, v_i), (\alpha_{i,1}, \ldots, \alpha_{i,t}, C_{i,1}, \ldots, C_{i,k}))$ can not be found on $\mathcal{F}_{\mathrm{BB}}$, then output $(\texttt{Verifier}, P_i, 0)$. Otherwise continue.
   (b) Compute $s_{i,j} = D_{x_{j,i}}(C_{i,j})$, and verify that $g^{s_{i,j}} = u_i \prod_{l=1}^{t} \alpha_{i,l}^{j^l}$. If so set $b_{j,i} = 1$, otherwise $b_{j,i} = x_{j,i}$. Hand $(\texttt{Write}, \texttt{Judgement}, P_i, b_{j,i})$ to $\mathcal{F}_{\mathrm{BB}}$.
   (c) Wait until $(\cdot, M_l, \texttt{Judgement}, P_i, b_{l,i})$ appears on $\mathcal{F}_{\mathrm{BB}}$ for $l = 1, \ldots, k$.
   (d) Do the following for $l = 1, \ldots, k$:
      i. If $b_{l,i} = 1$, then set $b'_{l,i} = 1$.
      ii. If $b_{l,i} \neq 1$ then check if $y_{l,i} = g^{b_{l,i}}$. If not set $b'_{l,i} = 1$. If so compute $s_{i,l} = D_{b_{l,i}}(C_{i,l})$, and verify that $g^{s_{i,l}} = u_i \prod_{\iota=1}^{t} \alpha_{i,\iota}^{l^\iota}$. If so set $b'_{l,i} = 1$ and otherwise set $b'_{l,i} = 0$.
   (e) If $\sum_{l=1}^{k} b'_{l,i} = k$ set $b = 1$ and otherwise 0. Then output $(\texttt{Verifier}, P_i, b)$.

Lemma 3. *Protocol 2 securely realizes the ideal functionality $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ in the $\mathcal{F}_{\mathrm{BB}}$-hybrid model with respect to $\mathcal{M}_{k/2}$-adversaries under the DDH-assumption in G_q.*

Each prover computes $2k + t$ full exponentiations in G_q. Each verifier computes 2 full exponentiations in G_q for each prover.

The above protocol differs from the original protocol of Feldman [13] in that it does not require any interaction from the prover. To achieve this each verifier must generate an El Gamal key for each prover.

In order to use a single key for each mix-server we would need a cryptosystem secure against adaptive Chosen Ciphertext Attacks (CCA-attacks) in the sense of Rackoff and Simon [42]. Cramer and Shoup [9] show that their cryptosystem is CCA-secure under the DDH-assumption, so there exists such a cryptosystem. There are two drawbacks of this approach. Firstly, the complexity of the prover increases. Secondly, a verifier is no longer able to verify the correctness of a false complaint, since the complaining verifier is unable to reveal its private key (revealing the key reveals the witnesses of honest provers). It can be shown that this variant is only secure for $\mathcal{M}_{n/3}$-adversaries.

If there are very few corrupted provers a combination of the two methods is possible. For the provers a CCA-secure cryptosystem is used, but instead of revealing the key to complain, a verifier proves the correctness of its claim in zero-knowledge to the other verifiers, but using a protocol similar to the above.

A CCA-secure cryptosystem that has the property that a decryptor can show directly to a third party the contents of a cryptotext without revealing its key would also solve the problem. Such a cryptosystem can be constructed under strong assumptions (cf. [4]).

4.2 A Realization of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ in the $\mathcal{F}_{\mathrm{BB}}$-Hybrid Model

Neff [33] and independently Furukawa and Sako [15] presents elegant ideas for proving the correctness of a shuffle, and Groth [23] recently refined the ideas of Neff [33], and gave a more rigorous analysis. Presently we are unable to transform any of these protocols into a UC-secure zero-knowledge proof without loosing the efficiency of the original protocol. The approach of Desmedt and Kurosawa [10] is better suited to the extraction requirements of the UC-framework, but their protocol allows malicious verifiers to make honest verifiers reject the "proof" of an honest prover. This means that their "proof" is not a realization of a proof of knowledge according to Functionality 4. They use global properties to avoid this difficulty, but in our modularized approach this is difficult. Furthermore, we need a proof of correctness of a decrypt shuffle instead of a re-encryption shuffle.

We construct a secure realization of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, using similar ideas as Desmedt and Kurosawa, that is practical if the number of verifiers is small, or if a strong majority of the mix-servers are honest. The following definition uses several different partitions of the verifiers such that there is one partition such that each block contains at least one honest verifier, and such that all partitions have the property that there is one block that contains no corrupt verifiers.

Definition 4 ((k,t)-set system). *Let $S = \{M_1, \ldots, M_k\}$ be a set. A (k,t)*-set system *is a family $F = \{T_1, \ldots, T_d\}$ of $(t+1)$-partitions $T_i = \{W_{i,1}, \ldots, W_{i,t+1}\}$ of S, such that $\forall A \subset S$, $|A| = t$, $\exists T \in F$ such that $\forall W \in T$ we have $W \not\subset A$.*

If there exists a (t,k)-set system, there exists such a set system with a minimal value of d. It is not hard to see that d grows exponentially with k when t/k is constant. However, if $k = (t+1)(t+1)$ any partition T_1 such that $|W_{1,j}| = t+1$ suffices, and for some practical values of k and t the value of d is not terribly large, i.e. (k,t)-set system can be found by brute force search. For example if $k = 10$, $t = 4$ then $F = \{T_0, \ldots, T_4\}$, where $T_i = \{\{j + i \mod 5, j + 5\}\}_{j=0}^{9}$, suffices and $d = 5$. More details on set systems can be found in [10].

Our protocol is based on a (k,t)-set system and works as follows. The prover constructs a chain $L = \overline{L}_0, \overline{L}_1, \ldots, \overline{L}_{t+1} = L'$ of lists and a list $(\alpha_1, \ldots, \alpha_{t+1})$ for each partition in the set system. The α_l:s are randomly chosen under the restriction $\prod_{l=1}^{t+1} \alpha_l = y$. The lists are randomly chosen under the restriction $((g, h\prod_{\iota=l+1}^{t+1} \alpha_\iota, \alpha_l, \overline{L}_{l-1}, \overline{L}_l), w_l) \in R_{\mathrm{DS}}$ for $l = 1, \ldots, t+1$. The witnesses w_l of the relations in a chain are encrypted with a semantically secure cryptosystem using the keys of the verifiers, and written on the bulletin board. The length of each chain is $t+1$, which ensures that t corrupted verifiers gets no information. The number of chains and how the links are revealed are determined by the (k,t)-set system in such a way that there is at least one chain in which all links are revealed to the set of honest verifiers. This ensures the immediate extraction required by the UC-framework.

Protocol 3 (Zero-Knowledge Proof of Correct Decrypt-Shuffle). The proof protocol $\pi = (P_1, \ldots, P_k, M_1, \ldots, M_k)$ consists of provers P_j, and verifiers

M_j. Let $t = B - 1$ and $F = \{T_1, \ldots, T_d\}$, where $T_i = \{W_{i,1}, \ldots, W_{i,t+1}\}$, be a (k,t)-set system known by all participants.

PROVER P_i.

1. Wait until $(\cdot, M_j, \texttt{Keys}, y_{j,1}, \ldots, y_{j,k})$ appears on $\mathcal{F}_{\mathrm{BB}}$ for $j = 1, \ldots, k$.
2. Wait for input $(\texttt{Prover}, (g, h, y, L, L'), (x, \pi, L_R))$.
3. Do the following for $\gamma = 1, \ldots, d$:
   (a) Set $\overline{L}_{\gamma,0} = L$, and $\overline{L}_{\gamma,t+1} = L'$.
   (b) Choose $a_{\gamma,1}, \ldots, a_{\gamma,t+1} \in \mathbb{Z}_q$ randomly under the restriction that $x = \sum_{l=1}^{t+1} a_{\gamma,l}$, and define $\alpha_{\gamma,l} = g^{a_{\gamma,l}}$, and $\beta_{\gamma,\iota} = \prod_{l=\iota}^{t+1} \alpha_{\gamma,l}$.
   (c) For $l = 1, \ldots, t$ choose a list $\overline{L}_{\gamma,l}$ randomly under the restriction that $((g, h\beta_{\gamma,l+1}, \alpha_{\gamma,l}, \overline{L}_{\gamma,l-1}, \overline{L}_{\gamma,l}), w_{\gamma,l}) \in R_{\mathrm{DS}}$ for some witness $w_{\gamma,l}$.
   (d) Let $w_{\gamma,t+1}$ be defined by $((g, h, \alpha_{\gamma,t+1}, \overline{L}_{\gamma,t}, \overline{L}_{\gamma,t+1}), w_{\gamma,t+1}) \in R_{\mathrm{DS}}$.
   (e) Compute $C_{\gamma,j} = E_{y_{j,i}}(w_{\gamma,l})$ where the relation between j and l is given by $j \in W_{\gamma,l}$, for $j = 1, \ldots, k$ and $l = 1, \ldots, t+1$.
4. Hand $(\texttt{Write}, \texttt{Proof}, \{\alpha_{\gamma,t+1}, \{\alpha_{\gamma,l}, \overline{L}_{\gamma,l}\}_{l=1}^{t}, \{C_{\gamma,j}\}_{j=1}^{k}\}_{\gamma=1}^{d})$ to $\mathcal{F}_{\mathrm{BB}}$.

VERIFIER M_j.

1. Generate El Gamal keys $(x_{j,i}, y_{j,i})$ for $i = 1, \ldots, k$, and hand $(\texttt{Write}, \texttt{Keys}, y_{j,1}, \ldots, y_{j,k})$ to $\mathcal{F}_{\mathrm{BB}}$.
2. On input $(\texttt{Question}, P_i, (g, h, y, L, L'))$ do:
   (a) If $(\cdot, P_i, \texttt{Proof}, \{\alpha_{\gamma,t+1}, \{\alpha_{\gamma,l}, \overline{L}_{\gamma,l}\}_{l=1}^{t}, \{C_{\gamma,j'}\}_{j'=1}^{k}\}_{\gamma=1}^{d})$ can not be found on $\mathcal{F}_{\mathrm{BB}}$ output $(\texttt{Verifier}, P_i, 0)$. Otherwise continue.
   (b) Do the following for $\gamma = 1, \ldots, d$:
      i. Set $\overline{L}_{\gamma,0} = L$, and $\overline{L}_{\gamma,t+1} = L'$.
      ii. Compute $w_{\gamma,l} = D_{x_{j,i}}(C_{\gamma,j})$, where l is defined by $j \in W_{\gamma,l}$.
      iii. Verify that $y = \prod_{\iota=1}^{t+1} \alpha_{\gamma,\iota}$, and $((g, h\beta_{\gamma,l+1}, \alpha_{\gamma,l}, \overline{L}_{\gamma,l-1}, \overline{L}_{\gamma,l}), w_{\gamma,l}) \in R_{\mathrm{DS}}$. If so, set $b_{j,\gamma} = 1$ and otherwise $b_{j,\gamma} = x_{j,i}$.
   (c) Hand $(\texttt{Write}, \texttt{Judgement}, P_i, \{b_{j,\gamma}\}_{\gamma=1}^{d})$ to $\mathcal{F}_{\mathrm{BB}}$
   (d) Wait until $(\cdot, M_{j'}, \texttt{Judgement}, P_i, \{b_{j',\gamma}\}_{\gamma=1}^{d})$ appears on $\mathcal{F}_{\mathrm{BB}}$ for $j' \neq j$.
   (e) Do the following for $\gamma = 1, \ldots, d$ and $j' = 1, \ldots, k$:
      i. If $b_{j',\gamma} = 1$, set $b'_{j',\gamma} = 1$.
      ii. If $b_{j',\gamma} \neq 1$, check if $b_{j',\gamma}$ is the private key corresponding to $y_{j',i}$. If not set $b'_{j',\gamma} = 1$. If so, compute $w_{\gamma,l} = D_{b_{j',\gamma}}(C_{\gamma,j'})$, where l is defined by $j' \in W_{\gamma,l}$, and check if $((g, h\beta_{\gamma,l+1}, \alpha_{\gamma,l}, \overline{L}_{\gamma,l-1}, \overline{L}_{\gamma,l}), w_{\gamma,l}) \in R_{\mathrm{DS}}$ and $y = \prod_{\iota=1}^{t+1} \alpha_{\gamma,\iota}$. If so set $b'_{j',\gamma} = 1$, and otherwise set $b'_{j',\gamma} = 0$.
   (f) If $\sum_{j'=1}^{k} \sum_{\gamma=1}^{d} b'_{j',\gamma} = kd$ then set $b = 1$ and otherwise $b = 0$. Then output $(\texttt{Verifier}, P_i, b)$.

Lemma 4. *Let $B \leq k/2$. Protocol 3 securely realizes the ideal functionality $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ in the $\mathcal{F}_{\mathrm{BB}}$-hybrid model with respect to $\mathcal{M}_B$-adversaries under the DDH-assumption in G_q.*

Each prover computes $O(5dtN)$ exponentiations in G_q, and each verifier computes $O(4dN)$ exponentiations in G_q for each prover.

In both Protocol 2 and 3 the El Gamal cryptosystem could be replaced by any semantically secure cryptosystem, under potentially stronger assumptions. Alternatively the ideal functionality for secure single message transmission could be used since each key is only used once, but that would require that the ideal functionality is altered to allow a receiver to "publish its private key".

The value of d, and the complexity of the protocol grows exponentially with the number of mix-servers if t/k is constant, but when the number of mix-servers is small, e.g. $k = 14$ and $t = 6$, or if there is a strong majority of honest mix-servers, e.g. $t = \sqrt{k}$, our scheme is practical.

5 Combining the Results

At this point combining the results to show that we have securely realized a universally composable mix-net is easy.

Theorem 1. *Let π be the composition of Protocol 1, with Protocol 2 and Protocol 3. Then π securely realizes $\mathcal{F}_{\mathrm{MN}}$ in the $(\mathcal{F}_{\mathrm{BB}}, \mathcal{F}_{\mathrm{KG}})$-hybrid model with respect to $\mathcal{M}_{k/2}$-adversaries under the DDH-assumption in G_q.*

Corollary 1. *The composition of π and π_{BB} securely realizes $\mathcal{F}_{\mathrm{MN}}$ in the $\mathcal{F}_{\mathrm{KG}}$-hybrid model with respect to* blocking *$\mathcal{M}_{k/2}$-adversaries under the DDH-assumption in G_q.*

Our mix-net is not "universally verifiable", i.e. an individual outsider can not verify the correctness of an execution. On the other hand nothing prevents the mix-servers to prove the correctness of a decrypt-shuffle to any set of outside verifiers such that the majority are honest. Furthermore, in some scenarios the assumption on the maximum number of corrupted mix-servers is well founded.

We require an ideal distributed key generation functionality. The natural next step is to try to find a protocol that securely realizes this functionality under various reasonable assumptions. Another interesting line of research is to find more efficient secure realizations of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ in various models. Scenarios where the number of mix-servers is large should also be considered.

6 Acknowledgments

I am grateful to Johan Håstad for his advice and support. I had discussions with Gunnar Sjödin. My discussions with Rafael Pass encouraged me to do this work. Andy Neff and Jun Furukawa kindly answered all my questions about their respective constructions. I also thank the anonymous referees for their advise.

References

1. M. Abe, *Universally Verifiable mix-net with Verification Work Independent of the Number of Mix-centers*, Eurocrypt '98, pp. 437-447, LNCS 1403, 1998.

2. M. Abe, *Flaws in Some Robust Optimistic Mix-Nets*, In Proceedings of Information Security and Privacy, 8th Australasian Conference, LNCS 2727, pp. 39-50, 2003.
3. D. Beaver, *Foundations of secure interactive computation*, Crypto '91, LNCS 576, pp. 377-391, 1991.
4. R. Canetti, *Towards realizing random oracles: Hash functions that hide all partial information*, Crypto '97, LNCS 1294, pp. 455-469, 1997.
5. R. Canetti, *Security and composition of multi-party cryptographic protocols*, Journal of Cryptology, Vol. 13, No. 1, winter 2000.
6. R. Canetti, *Universally Composable Security: A New Paradigm for Cryptographic Protocols*, http://eprint.iacr.org/2000/067 and ECCC TR 01-24. Extended abstract appears in 42nd FOCS, IEEE Computer Society, 2001.
7. D. Chaum, *Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms*, Communications of the ACM - CACM '81, Vol. 24, No. 2, pp. 84-88, 1981.
8. R. Canetti, Y. Lindell, R. Ostrovsky, A. Sahai, *Universally Composable Two-Party and Multi-Party Secure Computation*, 34th STOC, pp. 494-503, 2002.
9. R. Cramer, V. Shoup, *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack*, Crypto '98, pp. 13-25, LNCS 1462, 1998.
10. Y. Desmedt, K. Kurosawa, *How to break a practical MIX and design a new one*, Eurocrypt 2000, pp. 557-572, LNCS 1807, 2000.
11. D. Dolev, C. Dwork, M. Naor, *Non-Malleable Cryptography*, 23rd STOC, pp. 542-552, 1991.
12. T. El Gamal, *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469-472, 1985.
13. P. Feldman, *A practical scheme for non-interactive verifiable secret sharing*, In Proceedings of the 28th FOCS, pages 427-438, 1987.
14. A. Fujioka, T. Okamoto and K. Ohta, *A practical secret voting scheme for large scale elections*, Auscrypt '92, LNCS 718, pp. 244-251, 1992.
15. J. Furukawa, K. Sako, *An efficient scheme for proving a shuffle*, Crypto 2001, LNCS 2139, pp. 368-387, 2001.
16. J. Furukawa, H. Miyauchi, K. Mori, S. Obana, K. Sako, *An implementation of a universally verifiable electronic voting scheme based on shuffling*, Financial Cryptography '02, 2002.
17. R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, *Secure Distrubuted Key Generation for Discrete-Log Based Cryptosystems*, Eurocrypt '99, LNCS 1592, pp. 295-310, 1999.
18. O. Goldreich, S. Micali, and A. Wigderson, *How to Play Any Mental Game*, 19th STOC, pp. 218-229, 1987.
19. S. Goldwasser, L. Levin, *Fair computation of general functions in presence of immoral majority*, Crypto '90, LNCS 537, pp. 77-93, 1990.
20. S. Goldwasser, Y. Lindell, *Secure Multi-Party Computation Without Agreement*, In Proceedings of the 16th DISC, LNCS 2508, pp. 17-32, 2002.
21. S. Goldwasser, S. Micali, *Probabilistic Encryption*, Journal of Computer and System Sciences (JCSS), Vol. 28, No. 2, pp. 270-299, 1984.
22. P. Golle, S. Zhong, D. Boneh, M. Jakobsson, A. Juels, *Optimistic Mixing for Exit-Polls*, Asiacrypt 2002, LNCS, 2002.
23. N. Groth, *A Verifiable Secret Shuffle of Homomorphic Encryptions*, PKC 2003, pp. 145-160, LNCS 2567, 2003.
24. M. Jakobsson, *A Practical Mix*, Eurocrypt '98, LNCS 1403, pp. 448-461, 1998.

25. M. Jakobsson, *Flash Mixing*, In Proceedings of the 18th ACM Symposium on Principles of Distributed Computing - PODC '98, pp. 83-89, 1998.
26. M. Jakobsson, A. Juels, *Millimix: Mixing in small batches*, DIMACS Techical report 99-33, June 1999.
27. M. Jakobsson, A. Juels, *An optimally robust hybrid mix network*, In Proceedings of the 20th ACM Symposium on Principles of Distributed Computing - PODC '01, pp. 284-292, 2001.
28. Y. Lindell, A. Lysyanskaya, T. Rabin, *On the Composition of Authenticated Byzantine Agreement*, 34th STOC, pp. 514-523, 2002.
29. S. Micali, C. Rackoff, B. Sloan, *The notion of security for probabilistic cryptosystems*, SIAM Journal of Computing, Vol. 17, No. 2, pp. 412-426, 1988.
30. S. Micali, P. Rogaway, *Secure Computation*, Crypto '91, LNCS 576, pp. 392-404, 1991.
31. M. Michels, P. Horster, *Some remarks on a reciept-free and universally verifiable Mix-type voting scheme*, Asiacrypt '96, pp. 125-132, LNCS 1163, 1996.
32. M. Mitomo, K. Kurosawa, *Attack for Flash MIX*, Asiacrypt 2000, pp. 192-204, LNCS 1976, 2000.
33. A. Neff, *A verifiable secret shuffle and its application to E-Voting*, In Proceedings of the 8th ACM Conference on Computer and Communications Security - CCS 2001, pp. 116-125, 2001.
34. A. Neff, *Personal communication*, 2003.
35. V. Niemi, A. Renvall, *How to prevent buying of votes in computer elections*, Asiacrypt'94, LNCS 917, pp. 164-170, 1994.
36. W. Ogata, K. Kurosawa, K. Sako, K. Takatani, *Fault Tolerant Anonymous Channel*, Information and Communications Security - ICICS '97, pp. 440-444, LNCS 1334, 1997.
37. C. Park, K. Itoh, K. Kurosawa, *Efficient Anonymous Channel and All/Nothing Election Scheme*, Eurocrypt '93, LNCS 765, pp. 248-259, 1994.
38. T. Pedersen, *A threshold cryptosystem without a trusted party*, Eurocrypt '91, LNCS 547, pp. 522-526, 1991.
39. B. Pfitzmann, *Breaking an Efficient Anonymous Channel*, Eurocrypt '94, LNCS 950, pp. 332-340, 1995.
40. B. Pfitzmann, A. Pfitzmann, *How to break the direct RSA-implementation of mixes*, Eurocrypt '89, LNCS 434, pp. 373-381, 1990.
41. B. Pfitzmann, M. Waidner, *Composition and Integrity Preservation of Secure Reactive Systems*, 7th Conference on Computer and Communications Security of the ACM, pp. 245-254, 2000.
42. C. Rackoff, D. Simon, *Noninteractive zero-knowledge proofs of knowledge and chosen ciphertext attacks*, 22nd STOC, pp. 433-444, 1991.
43. K. Sako, J. Killian, *Reciept-free Mix-Type Voting Scheme*, Eurocrypt '95, LNCS 921, pp. 393-403, 1995.
44. C. Schnorr, M. Jakobsson, *Security of Signed El Gamal Encryption*, Asiacrypt 2000, LNCS 1976, pp. 73-89, 2000.
45. Y. Tsiounis, M. Yung, *On the Security of El Gamal based Encryption*, International workshop on Public Key Cryptography, LNCS 1431, pp. 117-134, 1998.
46. D. Wikström, *Five Practical Attacks for "Optimistic Mixing for Exit-Polls"*, to appear in proceedings of Selected Areas of Cryptography (SAC), LNCS, 2003.
47. D. Wikström, *A Universally Composable Mix-Net*, manuscript will be available at `http://eprint.iacr.org/`.

A Proofs

Because of space restrictions we are unable to present proofs of all claims in this conference version. We present shortened proofs of Lemma 2 and Lemma 4. Proofs of all claims are given in the full version of this paper [47].

Proof (Lemma 2). We describe an ideal adversary $\mathcal{S}(\cdot)$ that runs any hybrid adversary $\mathcal{A}' = \mathcal{A}^{(\mathcal{S}_{\mathrm{BB}}, \mathcal{S}_{\mathrm{KG}}, \mathcal{S}_{\mathrm{ZK}}^{R_{\mathrm{C}}}, \mathcal{S}_{\mathrm{ZK}}^{R_{\mathrm{DS}}})}$ as a black-box. Then we show that if $\mathcal{S}$ does not imply that the protocol is secure, then we can break the DDH-assumption.

THE IDEAL ADVERSARY $\mathcal{S}$. Let I_P and I_M be the set of indices of participants corrupted by $\mathcal{A}$ of the sender type and the mix-server type respectively. The ideal adversary $\mathcal{S}$ corrupts the dummy participants $\tilde{P}_i$ for which $i \in I_P$, and the dummy participants $\tilde{M}_i$ for which $i \in I_M$. The ideal adversary is best described by starting with a copy of the original hybrid ITM-graph $(V, E) = \mathcal{Z}'(\mathcal{H}(\mathcal{A}', \pi^{(\tilde{\pi}_1^{\mathcal{F}_{\mathrm{BB}}}, \tilde{\pi}_2^{\mathcal{F}_{\mathrm{KG}}}, \tilde{\pi}_3^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}}, \tilde{\pi}_4^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}})}))$, where $\mathcal{Z}$ is replaced by a machine $\mathcal{Z}'$.

The adversary $\mathcal{S}$ simulates all machines in V except those in $\mathcal{A}'$, and the corrupted machines P_i for $i \in I_P$ and M_i for $i \in I_M$ under $\mathcal{A}'$:s control. We now describe how each machine is simulated.

$\mathcal{S}$ simulates the machines P_i, $i \notin I_P$, and the ideal functionalities $\mathcal{F}_{\mathrm{BB}}$, $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ and $\mathcal{F}_{\mathrm{KG}}$ honestly. All M_j for $j \notin I_M$ are also simulated honestly, except for M_l, where l is chosen as the maximal index not in I_M, i.e. the last honest mix-server. The machine M_l plays a special role.

Simulation of Links $(\mathcal{Z}, \mathcal{A})$, $(\mathcal{Z}, P_i)$ for $i \in I_P$, and $(\mathcal{Z}, M_j)$ for $j \in I_M$. $\mathcal{S}$ simulates $\mathcal{Z}'$, $\tilde{P}_i$, for $i \in I_P$, and $\tilde{M}_j$ for $j \in I_M$, such that it appears as if $\mathcal{Z}$ and $\mathcal{A}$, $\mathcal{Z}$ and P_i for $i \in I_P$, and $\mathcal{Z}$ and M_j for $j \in I_M$ are linked directly. For details on this see [47].

Extraction from Corrupt Mix-Servers and Simulation of Honest Mix-Servers. When a corrupt mix-server M_j, for $j \in I_M$, writes $\texttt{Run}$ on $\mathcal{F}_{\mathrm{BB}}$, $\mathcal{S}$ must make sure that $\tilde{M}_j$ sends $(\texttt{Run})$ to $\mathcal{F}_{\mathrm{MN}}$. Otherwise it may not be possible to deliver an output to honest mix-servers. If an honest dummy mix-server $\tilde{M}_j$, for $j \notin I_M$, receives $(\texttt{Run})$ from $\mathcal{Z}$, $\mathcal{S}$ must make sure that M_j receives $(\texttt{Run})$ from $\mathcal{Z}'$. If an honest mix-server M_j, for $j \notin I_M$, outputs $(\texttt{Output}, L')$, $\mathcal{S}$ must make sure that $\tilde{M}_j$ does the same. This is done as follows.

1. Let $j \in I_M$. If $(\cdot, M_j, \texttt{Run})$ appears on $\mathcal{F}_{\mathrm{BB}}$ $\tilde{M}_j$ hands $(\texttt{Run})$ to $\mathcal{F}_{\mathrm{MN}}$. When $\mathcal{S}$ receives $(\mathcal{S}, \tilde{M}_j, \texttt{Run})$ or $((\mathcal{S}, \tilde{M}_j, \texttt{Output}, L'), \{(\tilde{M}_l, \tau_l)\}_{l=1}^k)$ from $\mathcal{C}_{\mathcal{I}}$ the simulation of $\mathcal{F}_{\mathrm{BB}}$ is continued.
2. Let $j \notin I_M$. If $\mathcal{S}$ receives $(\mathcal{S}, \tilde{M}_j, \texttt{Run})$ or $((\mathcal{S}, M_j, \texttt{Output}, L'), \{(M_l, \tau_l)\}_{l=1}^k)$ from $\mathcal{F}_{\mathrm{MN}}$, $\mathcal{Z}'$ hands $(\texttt{Run})$ to M_j.
3. Let $j \notin I_M$. If $\mathcal{Z}'$ receives $(\texttt{Output}, L')$ from M_j, $\mathcal{S}$ instructs $\mathcal{C}_{\mathcal{I}}$ to deliver $(\texttt{Output}, L')$ to $\tilde{M}_j$.

Extraction from Corrupt Senders and Simulation of Honest Senders. If a corrupt sender P_i, for $i \in I_P$, in the hybrid protocol produces a cryptotext and informs

$\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ such that its input is deemed valid, then $\mathcal{S}$ must make sure that this message is extracted and given as input to $\mathcal{F}_{\mathrm{MN}}$ by $\tilde{P}_i$.

When an honest dummy sender $\tilde{P}_i$, for $i \notin I_P$, receives a message m_i from $\mathcal{Z}$, $\mathcal{S}$ must ensure that P_i receives some message m_i' from $\mathcal{Z}'$. But $\mathcal{S}$ can not see m_i, and must therefore hand some other message $m_i' \neq m_i$ to P_i, and then later fix this flaw in the simulation before $\mathcal{A}'$ or $\mathcal{Z}$ notice it. This is done as follows.

1. Let $i \in I_P$. Until $\mathcal{S}$ receives $((\mathcal{S}, M_j, \texttt{Output}, L'), \{(M_l, \pi_l)\}_{l=1}^k)$ from $\mathcal{C}_{\mathcal{I}}$.
   (a) If $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ receives $(P_i, \texttt{Prover}, (g, y, u_i, v_i), r_i)$ such that $((g, y, u_i, v_i), r_i) \in R_{\mathrm{DS}}$, then consult the storage of $\mathcal{F}_{\mathrm{BB}}$ and look for a pair $(c, P_i, (u_i, v_i))$.
   (b) If $\mathcal{F}_{\mathrm{BB}}$ receives $(P_i, \texttt{Write}, (u_i, v_i))$ then look if $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ stored r_i under $(P_i, (g, y, u_i, v_i))$ such that $((g, y, u_i, v_i), r_i) \in R_{\mathrm{DS}}$.

   If such a pair $[(c, P_i, (u_i, v_i)), (P_i, (g, y, u_i, v_i), r_i)]$ is found then $\tilde{P}_i$ sends $m_i = v_i y^{-r_i}$ to $\mathcal{F}_{\mathrm{MN}}$ and ignores further such pairs. When $\mathcal{F}_{\mathrm{MN}}$ writes $(\tilde{P}_i, \texttt{Send})$ to $\mathcal{S}$, the simulation, of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ or $\mathcal{F}_{\mathrm{BB}}$ respectively, is continued.
2. Let $i \notin I_P$. When $\mathcal{S}$ receives $(\tilde{P}_i, \texttt{Send})$ from $\mathcal{F}_{\mathrm{MN}}$, then $\mathcal{Z}'$ sends a randomly chosen message $m_i' \in G_q$ to P_i.

How M_l and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ fix the flaw in the simulation. $\mathcal{S}$ must make sure that the faulty messages $m_i' \neq m_i$ introduced during simulation of honest senders, because it does not know the real messages m_i of the honest dummy participants $\tilde{P}_i$ for $i \in i_P$, are not noticed. This is done by modifying M_l and $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ as follows.

1. If $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ receives a tuple $(M_j, \texttt{Question}, M_l, (g, h_{l+1}, y_l, L_{l-1}, L_l))$ it verifies that a tuple on the form $(M_l, \texttt{Prover}, (g, h_{l+1}, y_l, L_{l-1}, L_l), \cdot)$ has been received. If so it sets $b = 1$ and otherwise $b = 0$. Finally it hands to $\mathcal{C}_{\mathcal{I}}$ $((\mathcal{S}, M_j, \texttt{Verifier}, M_l, (g, h_{l+1}, y_l, L_{l-1}, L_l), b), (M_j, \texttt{Verifier}, M_l, b))$.
2. M_l does the following instead of Step 7b in the protocol. Let $L' = \{m_i\}_{i=1}^{N'}$, and note that by construction $\mathcal{S}$ has received $((\mathcal{S}, M_j, \texttt{Output}, L'), \ldots)$, i.e. it knows L'. M_l chooses $r_{l,i} \in \mathbb{Z}_q$, and $\pi_l \in \Sigma_N$ randomly, and computes the list $L_l = \{(u_{l,i}, v_{l,i})\}_{i=1}^{N'} = \{(g^{r_{l,i}}, h_{l+1}^{r_{l,i}} m_{\pi_l^{-1}(i)}\}_{i=1}^{N'}$. Finally it hands $(\texttt{Prover}, (g, h_{l+1}, y_l, L_{l-1}, L_l), \cdot)$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, and $(\texttt{Write}, (\texttt{List}, L_l))$ to $\mathcal{F}_{\mathrm{BB}}$.

The first step ensures that $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ plays along with M_l and pretends to other M_j that M_l did prove his knowledge properly. The second step ensures that M_l fixes the flaw in the simulation introduced by $\mathcal{S}$ at the point when it did not know the messages sent by honest dummy participants $\tilde{P}_i$, for $i \notin I_P$.

This concludes the definition of the ideal adversary $\mathcal{S}$.

Reaching a Contradiction. Next we show, using a hybrid argument, that if the ideal adversary $\mathcal{S}$ defined above does not imply the security of Protocol 1, then we can break the DDH-assumption.

Suppose that $\mathcal{S}$ does not imply the security of the protocol. Then there exists a hybrid adversary $\mathcal{A}' = \mathcal{A}^{(\mathcal{S}_{\mathrm{BB}}, \mathcal{S}_{\mathrm{KG}}, \mathcal{S}_{\mathrm{ZK}}^{R_{\mathrm{C}}}, \mathcal{S}_{\mathrm{ZK}}^{R_{\mathrm{DS}}})}$, an environment $\mathcal{Z}$ with auxiliary input $z = \{z_n\}$, a constant $c > 0$ and an infinite index set $\mathcal{N} \subset \mathbb{N}$ such that for

$n \in \mathcal{N}$: $|\Pr[\mathcal{Z}_z(\mathcal{I}(\mathcal{S}, \tilde{\pi}^{\mathcal{F}_{\mathrm{MN}}})) = 1] - \Pr[\mathcal{Z}_z(\mathcal{H}(\mathcal{A}', \pi^{(\tilde{\pi}_1^{\mathcal{F}_{\mathrm{BB}}}, \tilde{\pi}_2^{\mathcal{F}_{\mathrm{KG}}}, \tilde{\pi}_3^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}}, \tilde{\pi}_4^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}})})) = 1]| \geq \frac{1}{n^c}$, where $\mathcal{S}$ runs $\mathcal{A}'$ as a black-box as described above, i.e. $\mathcal{S} = \mathcal{S}(\mathcal{A}')$.

Defining the Hybrids. Without loss we assume that $\{1, \ldots, N\} \backslash I_P = \{1, \ldots, \eta\}$, and define an array of hybrid machines $T_0, \ldots, T_\eta$. Set $T_0 = \mathcal{Z}_z(\mathcal{I}(\mathcal{S}(\mathcal{A}'), \tilde{\pi}^{\mathcal{F}_{\mathrm{MN}}}))$, and then define T_s by the following modification to T_0.

1. When $\mathcal{S}$ receives $(\tilde{P}_i, \texttt{Send})$ from $\mathcal{F}_{\mathrm{MN}}$, for $i \notin I_P$, it checks if $i \in \{1, \ldots, s\}$. If so it consults the storage of $\mathcal{F}_{\mathrm{MN}}$ to find the message m_i sent by $\tilde{P}_i$. Then $\mathcal{Z}'$ sends m_i to P_i. Otherwise $\mathcal{Z}'$ sends a random message $m_i' \in \mathbb{Z}_q$ to P_i.

By inspection of the constructions we see that the output of T_η is identically distributed to the output of $\mathcal{Z}_z(\mathcal{H}(\mathcal{A}', \pi^{(\tilde{\pi}_1^{\mathcal{F}_{\mathrm{BB}}}, \tilde{\pi}_2^{\mathcal{F}_{\mathrm{KG}}}, \tilde{\pi}_3^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}}, \tilde{\pi}_4^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}})}))$ since the only essential difference is that M_l does not hand knowledge of his transformation to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, but $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ ignores M_l's inability so this is not discovered by $\mathcal{A}$ or $\mathcal{Z}$.

If we set $p_s = \Pr[T_s = 1]$, we have $\frac{1}{n^c} \leq |p_0 - p_\eta| \leq \sum_{i=1}^{\eta} |p_{s-1} - p_s|$, which implies that there exists some fixed $0 < s \leq \eta$ such that $|p_{s-1} - p_s| \geq \frac{1}{\eta n^c} \geq \frac{1}{N n^c}$.

Defining a Distinguisher. We are now finally ready to define a distinguisher D.

D is confronted with the following test. An oracle first chooses $\alpha, \beta, \gamma \in \mathbb{Z}_q$ and a bit $b \in \{0,1\}$ randomly and defines $(y_l', u, v) = (g^\alpha, g^\beta, g^{b\alpha\beta + (1-b)\gamma})$. Then D is given (y_l', u, v) and the task is to guess b. D does the following. It replaces y_l in M_l:s key generation by y_l'. This does not change the distribution of this key and thus does not change any of the hybrids. Since M_l appears to behave honestly (with the help of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$), the fact that M_l does not know $\alpha = \log_g y_l'$ is never revealed, and since less than $k/2$ mix-servers are corrupted α need never be recovered. D then simulates T_s until P_s receives the message $(\texttt{Send}, m_s)$, at which point it forms $(u', v') = (u, u^{\sum_{j \neq l} x_j} v m_s)$. Then P_s is modified to hand $(\texttt{Write}, (u', v'))$ to $\mathcal{F}_{\mathrm{BB}}$, and $(\texttt{Prover}, (g, y, u', v'), 1)$ to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$. Furthermore, $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{C}}}$ is modified to a handle this message as if $((g, y, u', v'), 1) \in R_{\mathrm{C}}$, i.e. it will essentially lie on P_i's behalf. D then continues the simulation of T_s until it outputs a bit b', which is then output by D.

If (y_l', u, v) is a Diffie-Hellman triple, then (u', v') is a valid encryption of m_s and the output of D is identically distributed to the output of T_s. If on the other hand (y_l', u, v) is a random triple, then (u', v') corresponds to an encryption of a random message m_s', i.e. the output of D is identically distributed to T_{s-1}. We conclude that $|\Pr[D(g^\alpha, g^\beta, g^\gamma) = 1] - \Pr[D(g^\alpha, g^\beta, g^{\alpha\beta}) = 1]| = |p_{s-1} - p_s| \geq \frac{1}{N n^c}$, which contradicts the DDH-Assumption, and the theorem is true.

Proof (Lemma 4). We describe an ideal adversary $\mathcal{S}(\cdot)$ that runs any hybrid adversary $\mathcal{A}' = \mathcal{A}^{(\mathcal{S}_{\mathrm{BB}}, \mathcal{S}_{\mathrm{KG}})}$ as a black-box. Then we show that if $\mathcal{S}$ does not imply that the protocol is secure, then we can break the DDH-assumption.

THE IDEAL ADVERSARY $\mathcal{S}$. Let I_P and I_M be the set of indices of participants corrupted by $\mathcal{A}$ of the sender type and the mix-server type respectively. The ideal adversary $\mathcal{S}$ corrupts the dummy participants $\tilde{P}_i$ for which $i \in I_P$, and the dummy participants $\tilde{M}_j$ for which $j \in I_M$. The ideal adversary is best described by starting with a copy of the original hybrid ITM-graph $(V, E) =$

$\mathcal{Z}'(\mathcal{H}(\mathcal{A}', \pi^{\tilde{\pi}^{\mathcal{F}_{\mathrm{BB}}}}))$, where we have replaced $\mathcal{Z}$ by a machine $\mathcal{Z}'$. The adversary $\mathcal{S}$ simulates all machines in V except for those in $\mathcal{A}'$, and the corrupted machines P_i for $i \in I_P$ and M_i for $i \in I_M$ under $\mathcal{A}'$:s control. $\mathcal{S}$ simulates $\mathcal{F}_{\mathrm{BB}}$ honestly.

Simulation of Links $(\mathcal{Z}, \mathcal{A})$, $(\mathcal{Z}, P_i)$ for $i \in I_P$, and $(\mathcal{Z}, M_j)$ for $j \in I_M$. $\mathcal{S}$ simulates $\mathcal{Z}'$, $\tilde{P}_i$, for $i \in I_P$, and $\tilde{M}_j$ for $j \in I_M$, such that it appears as if $\mathcal{Z}$ and $\mathcal{A}$, $\mathcal{Z}$ and P_i for $i \in I_P$, and $\mathcal{Z}$ and M_j for $j \in I_M$ are linked directly. For details on this see [47].

Simulation of Honest Verifiers. When an honest verifier $\tilde{M}_j$, for $j \notin I_M$, receives $(\texttt{Question}, P_i, (g, h, y, L, L'))$ from $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$, $\mathcal{S}$ must ensure that the simulated honest verifier M_j receives $(\texttt{Question}, P_i, (g, h, y, L, L'))$ from $\mathcal{Z}'$. When the simulated honest verifier M_j hands $(\texttt{Verifier}, P_i, b)$ to $\mathcal{Z}'$, $\mathcal{S}$ must ensure that $(\texttt{Verifier}, P_i, b)$ is delivered to $\tilde{M}_j$. This is done as follows.

1. Let $j \notin I_M$. If $\mathcal{S}$ receives $((\mathcal{S}, \tilde{M}_j, \texttt{Verifier}, \tilde{P}_i, (g, h, y, L, L'), b), (\tilde{M}_j, \tau_j))$ from $\mathcal{C}_{\mathcal{I}}$, $\mathcal{Z}'$ hands $(\texttt{Question}, P_i, (g, h, y, L, L'))$ to M_j.
2. Let $j \notin I_M$. If $\mathcal{Z}'$ receives $(\texttt{Verifier}, P_i, b)$ from M_j, $\mathcal{S}$ hands $(1, \tau_j)$ to $\mathcal{C}_{\mathcal{I}}$, i.e. $\mathcal{S}$ instructs $\mathcal{C}_{\mathcal{I}}$ to deliver $(\texttt{Verifier}, \tilde{P}_i, b)$ to $\tilde{M}_j$.

Simulation of Honest Provers. If an honest dummy prover $\tilde{P}_i$, for $i \notin I_P$, receives a message $(\texttt{Prover}, (g, h, y, L, L'), w)$ from $\mathcal{Z}$, $\mathcal{S}$ must ensure that P_i constructs a simulated proof deemed valid by the verifiers M_j despite that $\mathcal{S}$ does not know w. To be able to do this $\mathcal{S}$ must ensure that the honest mix-servers M_j, for $j \notin I_M$, do not complain. This is done as follows.

1. Let $j \notin I_M$. M_j follows its program except that if $i \notin I_P$ it always sets $b_{j,\gamma} = 1$ in Step 2(b)iii (i.e. it never decrypts anything encrypted with $y_{j,i}$).
2. Suppose that $\mathcal{S}$ receives $(\mathcal{S}, \tilde{P}_i, \texttt{Prover}, (g, h, y, L, L'), 1)$ from $\mathcal{C}_{\mathcal{I}}$ for $i \notin I_P$. By construction there exists for each γ some partition $W_{\gamma,\zeta_\gamma} \cap I_M = \emptyset$.
   $\mathcal{S}$ hands $(\texttt{Prover}, (g, h, y, L, L'), \cdot)$ to P_i, where Step 3 in the program of P_i is replaced by the following. For $\gamma = 1, \ldots, d$:
   (a) Set $\overline{L}_{\gamma,0} = L$, and $\overline{L}_{\gamma,t+1} = L'$.
   (b) Choose $a_{\gamma,l} \in \mathbb{Z}_q$, for $l \neq \zeta_\gamma$, randomly and define $\alpha_{\gamma,l} = g^{a_{\gamma,l}}$ for $l \neq \zeta_\gamma$, $\alpha_{\gamma,\zeta_\gamma} = y(\prod_{l \neq \zeta_\gamma} \alpha_{\gamma,l})^{-1}$, and $\beta_{\gamma,\iota} = \prod_{l=\iota}^{t+1} \alpha_{\gamma,l}$.
   (c) For $l = 1, \ldots, \zeta_\gamma - 1$ choose a list $\overline{L}_{\gamma,l}$ randomly under the restriction that $((g, h\beta_{\gamma,l+1}, \alpha_{\gamma,l}, \overline{L}_{\gamma,l-1}, \overline{L}_{\gamma,l}), w_{\gamma,l}) \in R_{\mathrm{DS}}$ for some witness $w_{\gamma,l}$.
   For $l = t, \ldots, \zeta_\gamma$ choose a list $\overline{L}_{\gamma,l}$ randomly under the restriction that $((g, h\beta_{\gamma,l+2}, \alpha_{\gamma,l+1}, \overline{L}_{\gamma,l}, \overline{L}_{\gamma,l+1}), w_{\gamma,l+1}) \in R_{\mathrm{DS}}$ for some witness $w_{\gamma,l+1}$.
   (d) Define $w_{\gamma,\zeta_\gamma} = (1, 1, \ldots, 1)$.
   (e) Compute $C_{\gamma,j} = E_{y_{j,i}}(w_{\gamma,l})$ where the relation between j and l is given by $j \in W_{\gamma,l}$, for $j = 1, \ldots, k$ and $l = 1, \ldots, t + 1$.

Note that all components of the (corrupt) proof of P_i above except $C_{\gamma,j}$ for $j \in W_{\gamma,\zeta_\gamma}$ and $\gamma = 1, \ldots, d$ are identically distributed to the proof of a prover following its program.

Extraction from Corrupt Provers. If a corrupt prover P_i, for $i \in I_P$, constructs a valid proof of knowledge, $\mathcal{S}$ must extract the knowledge and forward it to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$.

$\mathcal{S}$ does this as follows. By construction there exists an $0 < \alpha \leq d$ and a list $M_{\omega_1}, \ldots, M_{\omega_{t+1}}$, such that $M_{\omega_l} \in W_{\alpha,l}$, and $M_{\omega_l} \notin I_P$ for $l = 1, \ldots, t+1$.

1. Suppose that $(\cdot, P_i, \texttt{Proof}, \{\alpha_{\gamma,t+1}, \{\alpha_{\gamma,l}, \overline{L}_{\gamma,l}\}_{l=1}^{t}, \{C_{\gamma,j}\}_{j=1}^{k}\}_{\gamma=1}^{d})$ for $i \in I_P$, appears on $\mathcal{F}_{\mathrm{BB}}$. $\mathcal{S}$ interrupts the simulation of $\mathcal{F}_{\mathrm{BB}}$ when $\mathcal{F}_{\mathrm{BB}}$ receives a message on the form $(\texttt{Write}, \texttt{Verifier}, P_i, \{b_{j,\gamma}\}_{\gamma=1}^{d})$ from M_j and such messages has been received from all other mix-servers.
   $\mathcal{S}$ then checks if the proof is deemed valid by the provers by performing the tests of Step 2(e)ii. If so $\mathcal{S}$ does the following.
   (a) It computes $w_{\alpha,l} = D_{x_{\omega_l,i}}(C_{\alpha,\omega_l})$ for $l = 1, \ldots, t+1$.
   (b) From $w_{\alpha,1}, \ldots, w_{\alpha,t+1}$ it is trivial to compute a witness w such that $((g, h, y, L, L'), w) \in R_{\mathrm{DS}}$.
   (c) Finally $\mathcal{S}$ hands $(\texttt{Prover}, (g, h, y, L, L'), w)$ to $\tilde{P}_i$ (who forwards it to $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$). When $\mathcal{S}$ receives $(\mathcal{S}, \tilde{P}_i, \texttt{Prover}, (g, h, y, L, L'), 1)$ from $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ it continues the simulation of $\mathcal{F}_{\mathrm{BB}}$.

Reaching a Contradiction. Next we show, using a hybrid argument, that if the ideal adversary $\mathcal{S}$ defined above does not imply that Protocol 3 is secure, then we can break the DDH-assumption.

Suppose that $\mathcal{S}$ does not imply the security of the protocol. Then there exists a hybrid adversary $\mathcal{A}' = \mathcal{A}^{\mathcal{S}_{\mathrm{BB}}}$, an environment $\mathcal{Z}$ with auxiliary input $z = \{z_n\}$, a constant $c > 0$ and an infinite index set $\mathcal{N} \subset \mathbb{N}$ such that for $n \in \mathcal{N}$: $|\Pr[\mathcal{Z}_z(\mathcal{I}(\mathcal{S}, \tilde{\pi}^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}})) = 1] - \Pr[\mathcal{Z}_z(\mathcal{H}(\mathcal{A}', \pi^{\tilde{\pi}^{\mathcal{F}_{\mathrm{BB}}}})) = 1]| \geq \frac{1}{n^c}$, where $\mathcal{S}$ runs $\mathcal{A}'$ as a black-box as described above, i.e. $\mathcal{S} = \mathcal{S}(\mathcal{A}')$.

Defining the Hybrids. Without loss we assume that $\{1, \ldots, N\} \backslash I_P = \{1, \ldots, \eta\}$. We define $T_0 = \mathcal{Z}_z(\mathcal{I}(\mathcal{S}(\mathcal{A}'), \tilde{\pi}^{\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}}))$, and then define T_s by the following modifications to T_0.

1. When $\mathcal{S}$ receives $(\texttt{Prover}, \tilde{P}_i, (g, h, y, L, L'), 1)$ for $i \notin I_M$, it checks if $i \in \{1, \ldots, s\}$. If so, $\mathcal{S}$ consults the internal storage of $\mathcal{F}_{\mathrm{ZK}}^{R_{\mathrm{DS}}}$ and finds the w stored under the tag $(\tilde{P}_i, (g, h, y, L, L'))$. Then it runs a P_i following the protocol on input $(\texttt{Prover}, (g, h, y, L, L'), w)$. If $i \notin \{1, \ldots, s\}$, then the simulation of P_i proceeds as outlined above.

By inspection of the constructions we see that T_η is identically distributed to $\mathcal{Z}_z(\mathcal{H}(\mathcal{A}', \pi^{\tilde{\pi}^{\mathcal{F}_{\mathrm{BB}}}}))$, since the only essential difference is that honest verifiers do not verify the proofs of honest provers, but this is never noticed by $\mathcal{A}'$ or $\mathcal{Z}$.

If we set $p_s = \Pr[T_s = 1]$, we have $\frac{1}{n^c} \leq |p_0 - p_\eta| \leq \sum_{s=1}^{\eta} |p_{s-1} - p_s|$, which implies that there exists some fixed $0 < s \leq \eta$ such that $|p_{s-1} - p_s| \geq \frac{1}{\eta n^c} \geq \frac{1}{N n^c}$.

Completing the Proof. We only argue informally for the remainder of the proof. For a formal proof we refer the reader to [47]. Informally we have shown that there is an adversary and an environment that can distinguish executions where the s:th prover follows its program and encrypts real shares of its proof, and executions where the s:th prover encrypts $(1, 1, \ldots, 1)$, for the honest verifiers. From this observation we construct a distinguisher. A hybrid argument shows that this distinguisher violates the DDH-assumption.