# Towards a Publicly-Verifiable Mix-Net Providing Everlasting Privacy

Johannes Buchmann[1] and Denise Demirel[1] and Jeroen van de Graaf[2]

[1] Technische Universität Darmstadt, Germany.
[2] Universidade Federal de Minas Gerais, Brazil.

**Abstract.** All implementations of verifiable mix-nets provide computational privacy only, because the audit information published is encrypted using some public key algorithm. Consequently, at some time in the future, when the underlying cryptographic assumption is broken, privacy is violated, and each output message can be traced back to its input. We address this problem by presenting a mix-net that uses a homomorphic, unconditionally hiding commitment scheme to encrypt the audit information, implying unconditional or everlasting privacy towards the public. The correctness of our mix-net is guaranteed with overwhelming probability even if all authorities conspire, under the assumption that the commitment scheme is computationally binding until the mixing process has ended. An implication of our result is that many current applications that use mix-nets can be upgraded to unconditional privacy.

**Keywords:** Mix-Net, Everlasting Privacy, Universal Verifiability

## 1 Introduction

### 1.1 Motivation

Mix-nets were introduced by David Chaum in 1981 [2] to allow anonymous communication within a network. Its basic functionality is to process a set of input messages, so that the content remains unchanged while any link between a single input and its associated output is removed. The reencryption mix-net introduced in 1993 by Park et al. [10] is based on this technique and allows that its correctness can be verified by any third party, i.e. showing that each message that comes in, goes out. Universally verifiable mix-nets are of interest for several applications where privacy plays an important role. Examples of real use are electronic auctions [6], electronic exam systems [7], and electronic voting [1, 12].

In all solutions for universal verifiable mix-nets that we know of, the audit information is encrypted using some public key algorithm, which is assumed to be computationally hard. However, when the underlying cryptographic assumption is broken (perhaps decades later) all the audit information can be decrypted and privacy is violated: each output message published can be traced back to its input. With current trends in technology, like quantum computers, such a scenario is realistic. With the cost for storing information becoming less, the

fact that (encrypted) information has been published on the internet makes it virtually impossible to remove this data later on. In addition, processing power increases continually following Moore's law. So all an attacker needs to do is to download the audit information published, wait until the cryptographic assumption is broken, and decrypt it. In other words, the privacy offered by current implementations of mix-nets has an (often unknown) expiration date.

### 1.2 High-level Description of our Result

In this paper we show how to use a homomorphic, unconditionally hiding commitment scheme to encode the audit information, i.e. the encoded messages and the proofs published for correct mixing. The system commits to a submitted message $t$ with a (randomly chosen) decommitment value $s$. Like with homomorphic encryption, each mix can recode (or rerandomize) a commitment $u = \mathsf{Com}(t, s)$ by multiplying it with $\mathsf{Com}(0, s')$, that is, $u' = \mathsf{Com}(t, s + s')$. Further, due to the homomorphic property of the commitment scheme the correctness of the rerandomization can be shown by proving knowledge of the used permutation and rerandomization value. However, in order to open shuffled commitments, one needs to know the decommitment values. Our solution is to send this data together with message $t$ as auxiliary information through a **private** mix-net to which the public has no access. Any rerandomization $u' = \mathsf{Com}(0, s')u$ has matching rerandomizations $\langle v', w' \rangle = \langle \mathsf{Enc}(0)v, \mathsf{Enc}(s')w \rangle$, where $v = \mathsf{Enc}(t)$ and $w = \mathsf{Enc}(s)$ and $\mathsf{Enc}$ is a suitable homomorphic encryption scheme. So essentially we use two tightly synchronized mix-nets run by the same mixes: one which is mixing commitments and which is fully public. And a second mix-net which uses homomorphic encryption to which the public has no access. Then, after the last mix $M_n$ has published its data, $v_n$ and $w_n$ are jointly decrypted yielding $s^* = s_0 + \cdots + s_n$ and $t$, the opening values of $u_n = \mathsf{Com}(t, s_0 + \cdots + s_n)$.

The scheme sketched in the above paragraphs already provides everlasting privacy towards observers. But it has one drawback: the first mix, to whom the users submit their message, gets to see $\mathsf{Enc}(t)$. So when, in some future, the encryption scheme gets broken, this mix, if dishonest, could reconstruct the user's message. If this is considered a problem, an obvious solution is to split the input in two (or more) parts, submit each part to a separate mix-net, and have the parts recombined, decoded, and published by a special publication authority.

### 1.3 Related Work

Despite the vast literature on mix-nets, we are not aware of any publication that accomplishes everlasting privacy for mixing. The protocol that comes closest to what we propose here is the shuffling used in the Split-Ballot voting system [9]. However, this protocol is devised for a very particular situation: two voting authorities who want to shuffle ballot shares of a specific format, allowing them to jointly compute the tally. It is not at all obvious how to utilize this shuffle protocol in other voting protocols, nor how to obtain a general, unconditional mix-net for mixing messages of arbitrary format. Another way to illustrate the difference

is as follows: in Split-Ballot, the voter shares a secret with two authorities "in parallel", whereas in conventional mixing a message is processed sequentially.

In [4] similar ideas presented here are directly applied on the Helios voting system. In independent work parallel to ours, Pereira et al. [11] approach the ideas of [4] and this paper from a different angle. The commitment scheme with matching encryption is presented as a new, unified primitive, and then used to show that any voting scheme which uses homomorphic tallying can have an unconditionally (or perfectly) private audit trail. A similar statement is made for voting protocols that use mix-nets, but, unlike this paper, no details of the mixing process are presented, nor does it address the issue of a mix-net providing everlasting privacy towards both the public and the mixing authorities.

The structure of this paper is straightforward: In Section 2 we show how the standard reencryption process can be adapted to provide everlasting privacy towards observers and present formal statements of the properties. Section 3 describes a mixing process providing everlasting privacy also towards the authorities, followed by conclusion and future work.

## 2 Mixing with Everlasting Privacy Towards the Public

### 2.1 Cryptographic Primitives

The protocol uses two cryptographic primitives. First, a commitment scheme in order to provide everlasting privacy and universal verifiability towards the public. Second, a matching encryption scheme allowing the authorities to open the commitment at the end of the mixing process and at the same time providing privacy towards the authorities.

A (non-interactive) **commitment scheme** is a triple $(\mathsf{GenCom}, \mathsf{Com}, \mathsf{Unv})$ such that $\mathsf{GenCom}(1^\kappa)$ generates the public commitment key $ck$ for security parameter $\kappa$. Note that the security parameter defines the message space $\mathcal{M}$ and the randomization space $\mathcal{R}$. We will suppose implicitly the presence of $ck$ in the remainder, leaving it out of the notation. $c = \mathsf{Com}(m, s) \in \mathcal{C}$ takes as input a message $m \in \mathcal{M}$, a uniformly chosen decommitment value $s \in \mathcal{R}$, resulting in a commitment $c \in \mathcal{C}$. The algorithm $\mathsf{Unv}(c, m, s)$ returns $m$ if $c = \mathsf{Com}(m, s)$ and $\perp$ if not. The commitment scheme has to provide the following properties. (1) Correctness: For any $m \in \mathcal{M}, r \in \mathcal{R} : \mathsf{Unv}(\mathsf{Com}(m, r), m, r) = m$. (2) Non-Interactive: All communication goes from the sender to the receiver. (3) Computationally Binding: Given a commitment $c = \mathsf{Com}(m, r)$, for any PPT $\mathcal{A}$ the probability to find a second opening pair $(m', r')$ with $m \neq m'$ such that $\mathsf{Com}(m, r) = \mathsf{Com}(m', r')$ is negligible in $\kappa$. (4) Unconditionally Hiding: For any pair $m, m' \in \mathcal{M}$ the distribution of the randomized values $\mathsf{Com}(m, r)$ and $\mathsf{Com}(m', r')$ must be identical when $r, r' \in \mathcal{R}$ are chosen uniformly random. (Obviously, this property can be weakened to statistically hiding, but for ease of exposition we do not explore this.) (5) Homomorphic: For all $m, m' \in \mathcal{M}$ and $r, r' \in \mathcal{R}$ $\mathsf{Com}(m, r) \cdot_{\mathcal{C}} \mathsf{Com}(m', r') = \mathsf{Com}(m +_{\mathcal{M}} m', r +_{\mathcal{R}} r')$. In the following, we will denote the neutral element $0_{\mathcal{M}}$ of $\mathcal{M}$ and $0_{\mathcal{R}}$ of $\mathcal{R}$ as 0. Furthermore, we will refer to the group operations $+_{\mathcal{M}}$ and $+_{\mathcal{R}}$ by $+$.

The **encryption scheme** $(\mathsf{GenEnc}, \mathsf{Enc}, \mathsf{Dec})$ used is identical to the one already used in standard reencryption mix-nets. The mixing process is based on a **homomorphic public key encryption algorithm** defined by the triple $(\mathsf{GenEnc}, \mathsf{Enc}, \mathsf{Dec})$ such that $\mathsf{GenEnc}$ generates two separate keys, a public key $pk$ and a private key $sk$, where the private key is presumed to be shared among a set of key trustees $\mathcal{T}$ using threshold decryption. $\mathsf{Enc}(t, s) = c$ denotes the encryption of message $t \in G$ with randomness $s \in H$ and public key $pk$. $\mathsf{Dec}(c) = t$ denotes the decryption of ciphertext $c$ to message $t \in G$ using private key $sk$. Note that the algorithm should provide semantical (CCA) security and be homomorphic in $t$ and in $s$, meaning that for all $t, t' \in G$ and for all $s, s' \in H$ : $\mathsf{Enc}(t, s) \cdot \mathsf{Enc}(t', s') = \mathsf{Enc}(t +_G t', s +_H s')$, where $+_G$ and $+_H$ are operations in group $G$ and $H$. As a consequence, reencrypting a ciphertext $c = \mathsf{Enc}(t, s)$ without knowing and changing message $t$ is possible by multiplying it with an encryption of the neutral element $0_G$ of group $G$, i.e. $\mathsf{ReEnc}(c, s') = \mathsf{Enc}(t, s) \cdot \mathsf{Enc}(0_G, s') = \mathsf{Enc}(t, s + s')$. However, because both the message $m \in \mathcal{M}$ and the auxiliary value $r \in \mathcal{R}$ have to be sent over the private channel, we need two instances of this encryption scheme. One, denoted as $E_{\mathcal{M}}$, which must be homomorphic over the message space $\mathcal{M}$. The second one, denoted as $E_{\mathcal{R}}$, whose message space is homomorphic over group $\mathcal{R}$.

An additional ingredient for reencryption mix-nets are **proofs of correct reencryption**: each mix has to provide a zero-knowledge proof that the data has been processed correctly, such that the set of output values is a valid shuffle of the set of input values. These proofs are made public, thus providing universal verifiability. In addition this mix-net uses **proofs of consistency**: each mix has to privately prove that the same permutation and random values have been used to rerandomize the published commitments and to reencrypt the corresponding private encrypted opening values. Note that both proofs have to be perfect zero-knowledge. More precisely, if a proof published by the prover is correct, even a computationally unbounded verifier cannot learn more than that.

To show the viability we now give some instantiations of a commitment scheme with matching encryption scheme satisfying the properties above. For their Split-Ballot voting system, Moran and Noar proposed the use of Paillier encryption in combination with a slightly modified Pedersen commitment Scheme [9, Appendix A]. We also believe that an alternative implementation with unconditional Jacobi symbols commitments and Rabin encryption might work, but the scheme would be very inefficient since each Jacobi symbol would implement only one bit. More interesting is the recent suggestion of Pereira et al. [11] for an efficient unconditionally hiding commitment scheme with matching homomorphic encryption scheme based on elliptic curves. If this could be used in combination with a non-interactive proof (or rather, argument) [5, 8] this might lead to a very efficient mixing procedure.

### 2.2 Assumptions

For a standard reencryption mix-net to be secure the following assumptions are made: **(A)** *The authorities cannot break the underlying computational problem of*

*the encryption scheme.* **(B)** *At least one mix is honest and keeps the association between its input and output values secret.* **(C)** *Using $(k, n)$-threshold decryption at least $(n-k+1)$ "key trustees" act honestly and keep their key portion secret.* **(D)** *All random challenge bits used in the mixing and verification steps comes from a trusted Random Beacon and are unpredictable.*

Apart from that, we make the following additional assumptions: **(E)** *The authorities cannot break the computational binding property of the commitment scheme for the parameters chosen before the messages have been published and certified.* **(F)** *There exists a private channel between the user and the first mix $M_1$ of the mix-net, between each mix and its successor in the mix-net, and between the last mix and the key trustees $\mathcal{T}$.* **(H)** *After the protocol has been certified all authorities destroy all information private to them.*

### 2.3 Adapted Mixing Process

For legibility, we will interpret a batch of messages each from a different user by a vector, denoted by a capital letter. Operations on the entries of vectors carry over to the vectors. For instance, the $\mathsf{Perm}$ operation permutes the entries of a vector: $T' = \mathsf{Perm}_\pi(T)$ means that $t'(i) = t(\pi(i))$.

**(I) Submission** To submit a message $t \in \mathcal{M}$ encoded with randomness $s$, the user calculates the triple $(u, v, w) = (\mathsf{Com}(t, s), \mathsf{Enc}_{\mathcal{M}}(t), \mathsf{Enc}_{\mathcal{R}}(s))$ and provides a proof of knowledge (POK) that the $t$ and $s$ used in all three components are the same. The user uses a private channel to send the triple $(u, v, w)$ to the system together with the POK for consistency. The input batch of the first mix consists of all triples received ordered in some canonical way and is denoted as $(U_0, V_0, W_0)$. The public part of the input batch, $U_0$, is published.

**(II) Mixing** We now describe the shuffling procedure for $K$ inputs and a mix-net consisting of $n$ mixes $M_1, M_2 \ldots, M_n$. The input batch of $M_j$ is defined as the output batch of the preceding mix $M_{j-1}$, except that the first mix receives its inputs directly from the system. In addition, the output batch of the last mix will be sent to the key trustees $\mathcal{T}$. In other respects, the shuffling procedure for each mix is identical. (1) Let the input batch of the mix $M_j$ be $(U_{j-1}, V_{j-1}, W_{j-1})$. Then $M_j$ rerandomizes the commitment vector: $U'_j = U_{j-1} \cdot \mathsf{Com}(0, S_j)$, it reencrypts the vector of encryptions of the contents: $V'_j = V_{j-1} \cdot \mathsf{Enc}_{\mathcal{M}}(0)$ and, through homomorphic encryption, updates the decommit value: $W'_j = W_{j-1} \cdot \mathsf{Enc}_{\mathcal{R}}(S_j)$. (2) To obtain the output batch, $M_j$ chooses a random permutation $\pi_j$ and sets $(U_j, V_j, W_j) = \mathsf{Perm}_{\pi_j}(U'_j, V'_j, W'_j)$. (3) The commitments $U_j$ are published, whereas the corresponding encryptions $V_j$ and $W_j$ are sent to $M_{j+1}$ through a private channel. (4) $M_j$ proves, in a publicly verifiable way, that $U_j$ is a recoding and permutation of $U_{j-1}$. (5) $M_j$ provides a POK to $M_{j+1}$ that the output batch is a consistent rerandomization and permutation of the entire input batch, i.e. it shows it knows permutation $\pi_j$ and the vector of random values $S_j$. Note that $\mathcal{T}$ verifies the output of the last mix.

**(III) Decoding and publication** The key trustees $\mathcal{T}$ compute and publish $T^* = \mathsf{Dec}(V_n)$ and $S^* = \mathsf{Dec}(W_n)$. Observe that due to the homomorphic properties and to the fact that the same permutations $\pi_j$ have been used in public

and the private network, we have that $T^*$ and $S^*$ are the values to open $U_n$, that is, $U_n = \mathsf{Com}(T^*, S^*)$.

**(IV) Certification** Auditors verify whether $U_n = \mathsf{Com}(T^*, S^*)$ and certify the output if this condition and all public proofs of knowledge hold.

### 2.4 Properties

For proving **correctness**, it is necessary to show that the mixes did not change any of the messages. In other words, $T_0 \equiv T_n$, where $\equiv$ stands for the existence of a permutation that maps $T_0$ to $T_n$. In our case correctness does not follow straightaway from existing proofs of mixing schemes since these are all based on encryption implying that the message $t$ is unambiguously defined. Since we want the correctness to be universally verifiable while preserving unconditional privacy, only public information can be used. However, as long as $\mathcal{T}$ can open the commitments that are published by the last mix, i.e. publish the values $T^*$ and $S^*$ such that $U_n = \mathsf{Com}(T^*, S^*)$, then $T_0 \equiv T_n$. In other words, we argue that for correctness it is completely irrelevant *how* $\mathcal{T}$ obtained $T^*$ and $S^*$. This is a consequence of the fact that the commitment scheme used is unconditionally hiding and computationally binding. This can be proven by contradiction, showing that if after running the protocol $T_0 \not\equiv T_n$, then there exists an efficient algorithm violating the binding property by computing a commitment $u$ that can be opened to two distinct values: $u = \mathsf{Com}(\tau_1, \sigma_1) = \mathsf{Com}(\tau_2, \sigma_2)$. Note that, because of the audit information published during all the mixing steps, any observer can perform the checks. This, together with the fact that the randomness of the challenge bit is guaranteed means that the protocol is **universally verifiable**. **Individual verifiability** follows, simply because of the fact that the users can check that their input $u_0$ is published.

If at least one mix is honest and keeps the used permutation secret, then **privacy** follows from the fact that the output batch $T_{out}$ is an unknown permutation of the input batch, $T_{in}$. Even a computationally unbounded attacker cannot obtain any additional information, since the commitments used to encode the messages are unconditional and all used proofs provide perfect zero-knowledge.

Regarding **robustness**, it is clear from the construction that if everybody is honest, then the process must succeed, always. If cheating is detected the malicious mix or the whole mix-net can simply be replaced. To avoid that a mix can lie about its private input towards the verifier, we ask the user and each mix to sign its output. In addition the users have to send a POK for consistency to ensure that they cannot submit inconsistent inputs without being detected.

## 3 Everlasting Privacy Towards the Authorities

Using the protocol presented in Section 2, the users submit the triple $(u, v, w)$, where $v$ is an encryption of the message $t$. Though they use a private channel, this protects them from outsiders, but not from the first mix, who will be able to recover $t$ once the computational assumption is broken. If this is unacceptable, a

possible solution is to have several mix-nets in parallel, have the users split their message, and submit each part to a different mix-net. As long as none of the first mixes nor any of their verifiers share their information with one mix of each mix-net, the privacy of the submitted message is guaranteed unconditionally.

A problem that arises here is how to match the various shares of one specific user that come out of the various mix-nets, needed to recombine the complete message. We solve this by supposing that a user submits each share labeled with the same, randomly chosen identification number $r_i$, which should be chosen large enough to avoid collision (say 128 bits). Since these $r_i$s cannot be public (to prevent any of the first mixes or its verifier from tracing back a message), the message $t$ and the random identity $r_i$ are encoded in separate commitments which are published and the decommitment values are sent together with the opening values of the corresponding messages towards the private mix-nets. After all data have been made anonymous by the mix-nets, the $r_i$s are decrypted by $\mathcal{T}$ and a new authority $\mathcal{Z}$ who keeps the obtained IDs private. The same $r_i$ should appear in each of the output batches of the various mixes and $\mathcal{Z}$ should therefore be able to match shares coming from the same user, reconstruct the message, and publish it. In addition, $\mathcal{Z}$ can use the published commitments to convince the public that the correct tuples were matched.

Correctness, Individual and Universal Verifiability can be shown similar to Section 2.4. The only difference is that $\mathcal{Z}$ has to prove that messages coming from the same user have been matched correctly. This is accomplished by a POK showing that when $\mathcal{Z}$ matches, shares coming from different mixes have the same $r_i$. With respect to privacy, we claim that under the additional assumption that the publication authority $\mathcal{Z}$ does not share its information with any of the first mixes or its verifier, nor do one of the first mixes or its verifier collaborate with one mix of each mix-net, privacy is also unconditional towards the authorities. But if they do share information then, as long as a sufficient number of "key trustees" are honest, they still need to break the encryption algorithm. For robustness, if all parties are honest then the protocol terminates successfully, unless two users chose the same $r_i$, which is an extremely unlikely event. Observe that when using secret sharing, a user can submit inconsistent IDs. There seems no obvious way to verify their consistency while maintaining unconditional privacy towards the first $i$ mixes, $M_{i,1}$. However, if users err, it just means their message cannot be decoded by $\mathcal{Z}$; it does not affect messages sent by other users. For applications where this can be caused by malicious software, a verification step during the submission phase could be implemented which prevents both, generating invalid messages and inconsistent IDs.

## Conclusion and Future work

We presented a novel protocol for a publicly-verifiable mix-net which offers everlasting privacy towards observers, meaning that the information made public does not help an adversary with unlimited computational resources. Further, we showed how the input can be split, each share mixed by a separate mix-net and

recombined providing everlasting privacy towards both, the observers and the authorities. We believe this is an important step forward since, as Chaum himself already argued in 1984 [3], individuals cannot be expect to understand the difference between computational and unconditional security, and they should not have to worry about it. In particular, computational security is simply not enough in some applications like elections, invalidating many proposed schemes. We plan to introduce this mix-net to mixing based voting systems like Prêt à Voter [12] or Split-Ballot [9]. Further, all known instantiations for a homomorphic commitment and encryption scheme are based on computational problems that are not quantum resistant. We plan to work on this matter in the future.

**Acknowledgement**

This work is a direct result of the third author's visit to Dagstuhl and Darmstadt in July 2011, and he would like to express his gratitude to the Dagstuhl Institute, to SnT at the University of Luxemburg, to CASED at the University of Darmstadt, and to his hosts, Peter Ryan and Johannes Buchmann, for making this visit possible. His research was partially supported by a FAPEMIG grant, number APQ-02719-10. Further, this paper has been developed within the project 'VerKonWa', funded by the DFG. Finally, we would like to thank Tal Moran, Olivier Pereira and Jeremy Clark for interesting discussions.

## References

1. Norwegian evote project, `http://www.regjeringen.no/en/dep/`
2. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM 24(2), pp. 84–88 (1981)
3. Chaum, D.: A new paradigm for individuals in the information age. In: IEEE Symposium on Security and Privacy. pp. 99–106 (1984)
4. Demirel, D., van de Graaf, J., Samarone dos Santos Araújo, R.: Improving helios with everlasting privacy towards the public. In: Proceedings of EVT/WOTE (2012)
5. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: ASIACRYPT. pp. 321–340 (2010)
6. Howlader, J., Ghosh, A., Pal, T.D.: Secure Receipt-Free Sealed-Bid Electronic Auction, p. 228 (2009)
7. Huszti, A., Pethö, A.: A secure electronic exam system. Publicationes Mathematicae Debrecen 77/3.-4., 299–312 (2010)
8. Lipmaa, H., Zhang, B.: A more efficient computationally sound non-interactive zero-knowledge shuffle argument. In: SCN. pp. 477–502 (2012)
9. Moran, T., Naor, M.: Split-ballot voting: Everlasting privacy with distributed trust. ACM Trans. Inf. Syst. Secur. 13(2) (2010)
10. Park, C., Itoh, K., Kurosawa, K.: Efficient anonymous channel and all/nothing election scheme. In: EUROCRYPT. pp. 248–259 (1993)
11. Pereira, O., Cuvelier, E., Peters, T.: Election verifiability or vote privacy: Do we need to choose? SecVote 2012 (2012), `http://secvote.uni.lu/`
12. Ryan, P.Y.A., Bismark, D., Heather, J., Schneider, S., Xia, Z.: Prêt à voter: a voter-verifiable voting system. IEEE TransIFS 4(4), 662–673 (2009)