Some remarks on a receipt-free and universally verifiable Mix-type voting scheme

Markus Michels · Patrick Horster

Theoretical Computer Science and Information Security
University of Technology Chemnitz-Zwickau,
Straße der Nationen 62, D-09111 Chemnitz, Germany
E-mail: {mmi,pho}@informatik.tu-chemnitz.de

Abstract. At Eurocrypt'95 Sako and Kilian presented the first Mix-type voting scheme which is receipt-free and universally verifiable.
In this contribution we analyze this scheme and show that the coercer must not collude with any center. Otherwise its robustness is lost. As a result, the assumed coercer model is clarified. More seriously, it is further pointed out that the privacy of votes can't be guaranteed, if only one Mix-center is honest. Hence, under the commonly used assumption that only one Mix-center must be honest, the voting scheme is insecure unless modified.

1 Introduction

Electronic voting is an interesting application of public key cryptography. Voting schemes have to satisfy several requirements, in particular (e.g. [FuOO92, BeTu94])

- *Completeness:* All valid votes are counted correctly, if all participants are honest.
- *Robustness:* Dishonest voters, other participants or outsiders can't disturb or disrupt an election.
- *Privacy:* The votes are casted anonymously.
- *Unreusability:* Every voter can vote only once.
- *Eligibility:* Only legitimate voters can vote.
- *Fairness:* A voter casts his vote independently and is not influenced (e.g. by publishing intermediate results of the election, copying and casting of the encrypted voting slip of another voter as his own vote).
- *Verifiability:* The tally can not be forged, as it can be verified by every voter. The verifiability is *local*, if a voter can only check if his own vote is counted correctly. If it is possible to verify whether all votes are counted correctly, then the verifiability is *universal*.
- *Receipt-freeness:* A voter can't prove to a coercer, how he has voted. As a result, verifiable vote buying is impossible.

Several schemes have been proposed, they can be divided in the class of schemes *without administrators* (e.g. [HuTe90, PfWa92]), schemes *with administrators*, which use *special encryption functions* to guarantee the anonymity of the votes (e.g. [BeYu86, Iver91, BeTu94, SaKi94, CFSY96]) and schemes with administrators which use *anonymous channels* to guarantee the anonymity of the votes. Within the latter class there are schemes based either on a *general* anonymous channel (e.g. [FuOO92, HoMP95]), a *special type* of an anonymous channel [Chau81, OkFO93]) or an anonymous channel based on *concrete* cryptographic mechanisms. The first voting based on such a channel was suggested by Park, Itoh and Kurusawa [PaIK93]. In particular, they used a Mix-net protocol with ElGamal encryption. A Mix-net [Chau81] consists out of n single Mix-centers and its goal is the realization of an anonymous channel. The input of the Mix-net is a number of messages. They will be transformed by the Mix-centers successively. The output of a Mix-net consists of the same messages but in a distinct order, in such a way that the link between a message and the sender of this message is unknown. It is usually assumed that only one of the Mix-centers must be honest. However, the Mix-net protocol proposed in [PaIK93] was shown to be vulnerable to some attacks [Pfit94] and there are further problems with the voting scheme as well [MiHo96]. At Eurocrypt'95, Sako and Kilian [SaKi95] suggested a voting scheme that is based on an improved Mix-net with ElGamal encryption. Receipt-freeness and universal verifiability are claimed to be gained.

The main contribution of this paper is the analysis of the used Mix-net protocol and the used coercer model in the scheme given in [SaKi95]. In particular, we point out that the mix-net is vulnerable to some known active and a new passive attack. While the active attacks can't be applied in the voting scheme environment, the new passive attack can. As a result, under the commonly used assumption that only one Mix-center must be honest, the privacy of the votes can't be guaranteed. Additionally, we discuss some coercer models and show that the coercer must not collude with *any* Mix-center. Otherwise the correctness of the tally can't be guaranteed.

In the next section we give a brief review of the used Mix-net protocol in [SaKi95], discuss some known attacks in section 3 and present a new attack in section 4. After that, we review the voting scheme due to Sako and Kilian. Finally, in section 6, we examine some attacks.

2 A Mix-net protocol

As already mentioned, the voting scheme presented in [SaKi95] is based on a modification of the Mix-net protocol presented by Park, Itoh and Kurosawa [PaIK93]. Although their original protocol was shown to be insecure [Pfit94], the modifications of the protocol should dwarf this attack. We give a short description of this Mix-net protocol here.

The primes p and q with $q|(p-1)$ and the generator g of order q are the public system parameters. Furthermore, $x_i \in \mathbf{Z}_q^*$ is the secret and $y_i := g^{x_i} \pmod p$ is the public key of Mix-center i. For simplicity, let $w_i := \prod_{j=i+1}^{n} y_j \pmod p$.

To shuffle a message m, any sender picks a random $r_0 \in \mathbf{Z}_q^*$ and transmits

$$Z_1 := (G_1, M_1) = (g^{r_0} \pmod p, w_0^{r_0} \cdot m \pmod p)$$

to the Mix-center 1. If Mix-center i $(1 \le i \le n-1)$ gets a list of values $Z_i := (G_i, M_i)$ (one value from each sender), then (for each value) he picks r_i at random, computes

$$G_{i+1} := G_i \cdot g^{r_i} \pmod p, M_{i+1} := M_i \cdot w_i^{r_i} / G_i^{x_i} \pmod p$$

and sends all values $Z_{i+1} = (G_i, M_i)$ in a shuffled order to Mix-center $i+1$. Mix-center n can recover all messages m by computing

$$m := M_n / G_n^{x_n} \pmod p.$$

To guarantee that he followed the protocol, each Mix-center i proves that he constructed $H_i := G_i^{x_i} \pmod p$ and Z_i correctly. The first is done by releasing H_i and a zero-knowlegde protocol given in [SaKi95] (called `DECRYPT`) and the latter by another zero-knowledge protocol given in [SaKi95] (called `SHUFFLE`). These proofs are not essential to understand our attacks later.

3 Discussion of known attacks

Clearly, the attack proposed by Pfitzmann [Pfit94] can still be applied, if only the Mix-net protocol is considered: If a dishonest sender Bob wants to find out what Alice sends and knows Alice's ciphertext (G_1, M_1) related to her message m, then he sends $(G_1^s \pmod p), M_1^s \pmod p))$ as his own ciphertext. If all messages are published on a public board, he computes $M^s \pmod p$ for every plaintext M in the public board and compares it with every plaintext. If there is a match, then $m = M$ with a high probability. As a result, in general the Mix-net protocol is not secure, although all Mix-centers might be honest. If the messages should satisfy a redundancy scheme (or are restricted to a few valid messages) the Mix-center n can refuse to publish Bob's message. But how should he prove to a third party, that his behavior is correct without leaking Bob's message? Furthermore, without loss of generality, we have to assume that the last Mix-center is dishonest. If he colludes with Bob, they can see together what Alice has sent, although Mix-center n might refuse to publish Bob's message.

Similar attacks were discussed in [SaKi95] concerning the problem of copying of votes. All these attacks have in common that Bob chooses his ciphertext in relation to Alice's ciphertext. However, as we will see later, the (encrypted) votes are not chosen arbitrarily by the voter, but are chosen out of two different possibilities given by the first Mix-center. If all communication is taped on a public board, then the attacks described above can't be applied in the voting scheme.

4 A new attack

We now investigate a new attack. More precisely, we assume that all Mix-centers except the jth's collude, but a colluding participant is not needed. Furthermore, the colluders need not to derivate from the protocol. Thus the attack is *passive* in contrary to the attacks mentioned in the last section. Alice sends

$$(G_1, M_1) := (g^{r_0} \pmod p), m \cdot \prod_{t=1}^{n} y_i^{r_0} \pmod p))$$

to Mix-center 1, together with her identity. Then the message is sent through the Mix-net as described above. Mix-center $j-1$ sends

$$(G_j, M_j) := (g^{r_0+R} \pmod p), m \cdot \prod_{t=j}^{n} y_t^{r_0+R} \pmod p))$$

to the honest Mix-center j, where $R \equiv \sum_{t=1}^{j-1} r_t \pmod q$. Note that Mix-center j reveals $H_j := G_j^{x_j} \pmod p$. Then all colluding Mix-centers can compute

$$\begin{aligned} M := M_j / \left(H_j \cdot \prod_{t=j+1}^{n} G_j^{x_t} \right) &\equiv m \cdot \left(\prod_{t=j}^{n} y_t^{r_0+R} \right) \cdot G_j^{-x_j} \cdot \prod_{t=j+1}^{n} G_j^{-x_t} \\ &\equiv m \cdot \left(\prod_{t=j}^{n} y_t^{r_0+R} \right) \cdot y_j^{-r_0-R} \cdot \prod_{t=j+1}^{n} y_t^{-r_0-R} \equiv m \pmod p \end{aligned}$$

together. Note that the first $j-1$ Mix-centers are necessary to know that (G_j, M_j) is related to Alice ciphertext and the last $n-j$ Mix-centers are important for the computation above, as their secret key is used. We will see later that this attack can be applied in the voting scheme.

5 The voting scheme

We describe the voting scheme given in [SaKi95] now: The initialization of the scheme is based on the initialization of the Mix-net protocol described above. Additionally, each voter i possesses a key pair $(a_i, \alpha_i := g^{a_i} \pmod p))$ for a chameleon bit commitment scheme [BrCC88] that is used for committing random strings. This is needed here for proving the order of the votes to the voter and gives the voter the opportunity to "prove" to the coercer whatever he wants. Furthermore, the election is restricted to a "yes/no" vote. The votes which can be casted are fixed, m_1 for the "yes", m_0 for the "no"-vote.

1. First, the last Mix-center n commits a random string $\pi^{(i,n)}$ of length $l+1$ bit to voter i using the public key α_i. He picks random $r_{2n-1}, r_{2n} \in \mathbf{Z}_q^*$, computes with $w_0 := \prod_{i=1}^{n} y_i \pmod p$

   $$\begin{aligned} v_0 &:= (\bar{G}_n, \bar{M}_n) = (g^{r_{2n}} \pmod p), m_0 \cdot w_0^{r_{2n}} \pmod p)) \\ v_1 &:= (\bar{G}'_n, \bar{M}'_n) = (g^{r_{2n-1}} \pmod p), m_1 \cdot w_0^{r_{2n-1}} \pmod p)) \end{aligned}$$

   and places (v_0, v_1) if the first bit of $\pi^{(i,n)} = 0$ and (v_1, v_0) otherwise. He proves in zero-knowledge (using the protocol `prove 1-0 vote` in [SaKi95]) that v_0, v_1 are constructed properly. The random string $\pi^{(i,n)}$ (except the first bit) is used in this zero-knowledge protocol.

2. The Mix-center n decommits $\pi^{(i,n)}$ through an untappable channel to the voter. Then, only the voter knows the order of v_0, v_1.
3. The next Mix-center j ($j = n-1, \ldots, 1$) gets $((\bar{G}_{j+1}, \bar{M}_{j+1}), (\bar{G}'_{j+1}, \bar{M}'_{j+1}))$ or $((\bar{G}'_{j+1}, \bar{M}'_{j+1}), (\bar{G}_{j+1}, \bar{M}_{j+1}))$ from Mix-center $j+1$ for each voter i, commits a $l+1$ bit string $\pi^{(i,j)}$ to the voter and picks random $r_{2j-1}, r_{2j} \in \mathbf{Z}_q^*$. In the first case, he computes
$$(\bar{G}_j, \bar{M}_j) := (\bar{G}_{j+1} \cdot g^{r_{2j}} \pmod p, \bar{M}_{j+1} \cdot w_0^{r_{2j}} \pmod p)$$
$$(\bar{G}'_j, \bar{M}'_j) := (\bar{G}'_{j+1} \cdot g^{r_{2j-1}} \pmod p, \bar{M}'_{j+1} \cdot w_0^{r_{2j-1}} \pmod p)$$
and in the second case he computes
$$(\bar{G}_j, \bar{M}_j) := (\bar{G}'_{j+1} \cdot g^{r_{2j}} \pmod p, \bar{M}'_{j+1} \cdot w_0^{r_{2j}} \pmod p)$$
$$(\bar{G}'_j, \bar{M}'_j) := (\bar{G}_{j+1} \cdot g^{r_{2j-1}} \pmod p, \bar{M}_{j+1} \cdot w_0^{r_{2j-1}} \pmod p).$$
Then he places the pairs as $((\bar{G}_j, \bar{M}_j), (\bar{G}'_j, \bar{M}'_j))$ if the first bit of the bitstring is one and otherwise in reverse order. Using `prove SHUFFLE` he can prove that he followed the protocol correctly by choosing λ in each round depending on the first unused bit of the bitstring. Note that there are only two choices for λ, because only two orders are possible. Therefore λ is completely determined by the random bit of the bitstring.
4. By decommitting the bitstring, the order of the Mix-centers' output is revealed only to the voter.
5. The steps 2 and 3 are repeated for every Mix-center. Then there are $(z_1, \ldots, z_t) \in \{0,1\}^t$, such that the voter gets the pairs
$$(\bar{G}_1, \bar{M}_1) = (g^{\sum_{t=1}^{n} r_{2 \cdot t - z_t}} \pmod p, m_0 \cdot w_0^{\sum_{t=1}^{n} r_{2 \cdot t - z_t}} \pmod p)$$
$$(\bar{G}'_1, \bar{M}'_1) = (g^{\sum_{t=1}^{n} r_{2 \cdot t + z_t - 1}} \pmod p, m_1 \cdot w_0^{\sum_{t=1}^{n} r_{2 \cdot t + z_t - 1}} \pmod p).$$
6. The voter knows which pair is which vote and posts his choice to the Mix-center 1. Now this pair is sent through the Mix-net to the Mix-center n, who is able to recover the voter's vote. Note that the form of one pair can be interpreted as the encrypted vote by the voter with "random" $\sum_{t=1}^{n} r_{2 \cdot t - z_t} \pmod q$ or $\sum_{t=1}^{n} r_{2 \cdot t + z_t - 1} \pmod q$. Therefore, it can be easily sent through the Mix-net protocol described above.
7. The last center reveals the votes and everybody can compute the tally.

6 Examining some attacks

6.1 Coercer model

The model of a coercer depends on the number of administrators in the voting scheme. Here we focus on a scheme with several administrators, each Mix-center can be regarded as an administrator. We can distinguish several abilities the coercer might have to control the voter. It might be called level of *voter-control.*

1. The coercer knows all public values only. He is not able to see physically what the voter sends as his vote.
2. The coercer knows all public values. Furthermore he sees what the voter sends as his vote, but there are possibilities during the election for the voter to communicate with the administrators privately.
3. The coercer knows all public values. Furthermore he sees what the voter sends as his vote, but there are possibilities during the election for the administrators to send private information to the voter.
4. The coercer knows all public values and he controls all communications of the voter during the election. Only in the initialization of the system the voter can communicate with the administrators privately.
5. The coercer knows all public values and controls all communications between the voter and the administrators.

Furthermore it can be distinguished with whom the coercer might collude and form a coalition. This might be called *level of coalition*. Possible colluders are

1. other voters,
2. at least one administrator
3. all administrators

In the protocol given in [SaKi95] the coercer has voter-control level 3 and coalition level 1 abilities: Assume that the one center and the coercer colludes. The coercer forces the voter i to reveal his secret key a_i, which he then posts to the dishonest Mix-center. This Mix-center can forge the decommitment of the random string and therefore the way, how he has shuffled. As a result, it's possible that the voter's vote is counted wrongly. Although the cheaters can't determine the result according to their wishes, this influences the output of the election and violates the robustness of the voting scheme. Therefore, it must be assumed that the coercer does not collude with *any* Mix-center.

6.2 Unsuccessful attack on the privacy of casted votes

Assume that the voter Bob, the first and the last center collude. Their goal is to find out how Alice has voted. If Bob knows Alice's ciphertext, he can transform this as described in section 3 (exponent by a random s) and sends it through the Mix-net instead of his vote. Clearly, the first center will notice that, as he still knows the two pairs he has posted to Bob (which are different from the ciphertext he gets) and the last center can see that Bob's vote is neither a "yes" nor a "no" vote. By computing the s-root of Bob's vote, it is easily computable how Alice has voted. However, if the first center is forced to accept just one of the two votes he has sent to the voters (e.g. everything is written on a public board), then this attack fails.

6.3 Successful attack on the privacy of casted votes

Assume that only Mix-center j is honest, while the other Mix-centers are dishonest and collude. Again, the goal is to find out how Alice has voted. All Mix-centers prepare two votes for Alice,

$$(g^{r_0} \pmod p), m_0 \cdot w_0^{r_0} \pmod p)), (g^{\bar{r}_0} \pmod p), m_1 \cdot w_0^{\bar{r}_0} \pmod p))$$

and r_0 and $\bar{r}_0$ is the sum of the used r's of all Mix-centers. Now Alice sends one of the votes back to Mix-center 1, say (G_1, M_1), which is published together with Alice's identity on the public board. This is necessary to avoid the forgery mentioned above. Then the attack described in section four can be applied. Thus the colluding Mix-centers can find out how Alice has voted. As a result, the privacy property is violated, if only one Mix-center is honest and the scheme is insecure under the commonly used assumption that only one Mix-center must be honest.

7 Conclusion

We reviewed the Mix-net protocol given in [SaKi95] and showed its vulnerability to some known and a new attack. Then we described the voting scheme given in [SaKi95] briefly. We have presented several coercer models and pointed out that the coercer must not collude with any Mix-center. This fact was independently observed by Gennaro, Jakobsson and Schoenmakers. Furthermore, we showed that the privacy of votes isn't guaranteed in the voting scheme described in [SaKi95], if only one Mix-center is honest. This results in the insecurity of the voting scheme under the commonly used assumption, that only one Mix-center must be honest. It seems that the attack can be prevented if at least two Mix-centers are assumed to be honest and therefore – under this assumption – the scheme might remain secure. After being informed by the authors, Sako suggested a modification of the used Mix-net protocol that seems to avoid this weakness, even if only one Mix-center is honest [Sako95].

8 Acknowledgments

The authors would like to thank Rossario Gennaro, Markus Jakobsson, Berry Schoenmakers and Tobias Zimmer for helpful comments.

References

[BeTu94] J.C.Benaloh and D.Tuinstra, "Receipt-Free Secret-Ballot Elections", Symposium on the Theory of Computing'94, (1994), pp. 544–53.

[BeYu86] J.C.Benaloh, M.Yung, "Distributing the Power of a Government to Enhance the Privacy of Voters", Proc. of Symposium of Principles of Distributed Computing, (1986), pp. 52–62.

[BrCC88] G.Brassad, D.Chaum, C.Crepéau, "Minimum disclosure proofs of knowledge", Journal of Computer and System Science, Vol. 37, (1988), pp. 156–189.

[Chau81] D.Chaum, "Untraceable electronic mail return addresses and digital pseudonyms", Communcations of the ACM, Vol. 24, No. 2, Feb., (1981), pp. 84–88.

[CFSY96] R.J.F.Cramer, M.Franklin, L.A.M.Schoenmakers, M.Yung, "Multi-authority secret-ballot elections with linear work", Lecture Notes in Computer Science 1070, Advances in Cryptology: Proc. Eurocrypt'96, Springer Verlag, (1996), pp. 72–83.

[FuOO92] A.Fujioka, T.Okamoto, K.Ohta, "A practical secret voting scheme for large scale elections", Lecture Notes in Computer Science 718, Advances in Cryptology: Proc. Auscrypt'92, Springer Verlag, (1992), pp. 244–51.

[HoMP95] P.Horster, M.Michels, H.Petersen, "Blind Multisignatures and their relevance for electronic voting", Proc. 11th Annual Computer Security Applications Conference, IEEE–Press, (1995), pp. 149–156.

[HuTe90] M.Huang, S.Teng, "Security, Verifiability, and Universality in Distributed Computing", Journal of Algorithm, Vol. 11, (1990), pp. 492–521.

[Iver91] K.R.Iversen, "A Cryptographic Scheme for Computerized General Elections", Lecture Notes in Computer Science 576, Advances in Cryptology: Proc. Crypto'91, Springer Verlag, (1992), pp. 405–419.

[MiHo96] M.Michels, P.Horster, "Cryptanalysis of a voting scheme", Proc. Communications and Multimedia Security II, Chapman & Hall, (1996).

[OkFO93] T.Okamoto, A.Fujioka, K.Ohta, "A practical secret voting scheme without anonymous channels", Symposium on Cryptography and Information Security, SCIS 93-1C, (1993), 12 pages.

[PaIK93] C.Park, K.Itoh, K.Kurosawa, "All/Nothing Election Scheme and Anonymous Channel", Lecture Notes in Computer Science 765, Advances in Cryptology: Proc. Eurocrypt'93, Springer Verlag, (1994), pp. 248–259.

[Pfit94] B.Pfitzmann, "Breaking an efficient anonymous channel", Lecture Notes in Computer Science 950, Advances in Cryptology: Proc. Eurocrypt'94, Springer Verlag, (1995), pp. 332–340.

[PfWa92] B.Pfitzmann, M.Waidner, "Unconditionally Untraceable and Fault-tolerant Broadcast and Secret Ballot Election", Hildesheimer Informatik-Berichte, Department of Computer Science, University of Hildesheim, Germany, May, (1992).

[Sako95] K.Sako, "An improved universally verifiable mix-type voting scheme", manuscript, October, (1995), 6 pages.

[SaKi94] K.Sako, J.Kilian, "Secure Voting Using Partially Compatible Homomorphisms", Lecture Notes in Computer Science 839, Advances in Cryptology: Proc. Crypto'94, Springer Verlag, (1994), pp. 411–424.

[SaKi95] K.Sako, J.Kilian, "Receipt-Free Mix-Type voting scheme", Lecture Notes in Computer Science 921, Advances in Cryptology: Proc. Eurocrypt'95, Springer Verlag, (1995), pp. 393–403.