5

Cryptanalysis of a 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'93 Park, Itoh and Kurusawa presented a voting scheme based on an efficient concrete mix-net. However, Pfitzmann pointed out that the used concrete mix-net is vulnerable to active and passive attacks. Therefore, the anonymity of the votes can't be guaranteed. Furthermore, Pfitzmann discussed some countermeasures against the attacks and how far they help.

In this paper we point out that *independent* of the used mix-net the voting scheme suffers from even more weaknesses. More precisely, we show that neither robustness nor fairness can be gained, although the main purpose of this voting scheme was to guarantee fairness.

Keywords

cryptology, cryptanalysis, voting schemes, anonymous channels, mix-nets

1 INTRODUCTION

One interesting application of public key cryptography is electronic voting. Voting schemes should satisfy several requirements, namely (see e.g. Fujioka, Okamoto and Ohta, 1992, Benaloh and Tuinstra, 1994)

- *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 vote slip of another voter as his own vote).

P. Horster (ed.), *Communications and Multimedia Security II*

- *Verifiability:* The tally can not be forged, as it can be verified by every voter. The verifiability is *locally*, if a voter can only check if his own vote if counted correctly. If it is verifiable whether all votes are counted correctly, then the verifiability is *universally.*
- *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, roughly they can be divided in the class of schemes without administrators, schemes with administrators, which uses anonymous channels to guarantee the anonymity of the vote and schemes with administrators which use special encryption functions to guarantee the anonymity of the votes. For a more detailed classification and an overview about various voting schemes see Horster and Michels (1995).

The voting scheme presented by Park, Itoh and Kurusawa (1993) belongs to the class of schemes with administrators and anonymous channels. Pfitzmann (1994) pointed out that the used concrete mix-net is vulnerable to active and passive attacks. Thus the anonymity of the votes can't be guaranteed unless the underlying mix-net is modified.

In this paper we point out that *independent* of the used mix-net the voting scheme suffers from even more weaknesses. In fact, neither robustness nor fairness can be gained, although the main purpose of this voting scheme was to guarantee fairness.

We review the concrete mix-net protocols, the known attacks and countermeasures in section 2, the voting scheme by Park, Itoh and Kurusawa in section 3 and cryptanalyse it in section 4. In section 5 we discuss some countermeasures.

2 TWO CONCRETE MIX-NET PROTOCOLS

A mix-net consists out of n single mix-centers. The goal of a mix-net 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 sended message and the sender of this message is unknown. It is usually assumed that only one of the mix-centers is honest.

We now review the concrete mix-net protocols by Park, Itoh and Kurusawa (1993), which is based on the public-key encryption scheme due to ElGamal (1985).

2.1 Common beginning

Let p a large prime, $q|(p-1)$ and g a generator of order q be the public system parameters, $x_i \in \mathbf{Z}_q^*$ the secret, $y_i := g^{x_i} \pmod p$ the public parameter of mix i. For simplicity, let $w_i := \prod_{j=i+1}^{n} y_j \pmod p$.

To shuffle a message $m \in \mathbf{Z}_{p-1}$, any sender picks a random r_0 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.

2.2 First protocol

If mix-center i ($1 \leq i \leq n-1$) 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 $i+1$. Finally, mix-center n can recover all messages m by computing

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

2.3 Second protocol

This protocol consists out of two steps:

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

   $$G_{i+1} := G_i \cdot g^{r_i} \pmod{p}, M_{i+1} := M_i \cdot w_0^{r_i} \pmod{p}$$

   and sends all values $Z_{i+1} = (G_i, M_i)$ in a shuffled order to mix-center $i+1$ or, if $i = n$, then all values $Z_n = (G_n, M_n)$ are written in a list.
2. Each mix-center i adds $H_i := G_n^{x_i} \pmod{p}$ to each value $Z_n = (G_n, M_n)$ in the list. Then everybody can compute the messages by

   $$m := M_n / (\prod_{i=1}^{n} H_i) \pmod{p}$$

2.4 Discussion of known attacks

It was pointed out by Pfitzmann (1994) that both concrete mix-nets are vulnerable to a simple passive attack, if $q = p - 1$ is chosen, as suggested by Park, Itoh and Kurusawa (1993). Under reasonable circumstances it's possible to link a ciphertext Z_1 to the related message m. However, if q is prime this attack can be avoided.

Furthermore, both mix-nets are susceptible to the following active attack (Pfitzmann, 1994): A dishonest mix-net reveals Z_1 of a honest sender to another dishonest sender. He sends

$$Z_1' := (G_1^l \pmod{p}, M_1^l \pmod{p})$$

through the mix-net. If the messages are processed as described, then the message of the honest sender m and the message of the dishonest sender $m^l \pmod{p}$ are part of the output later. As the attackers know l, they can compute $e^l \pmod{p}$ with every entry e of the output list of the mix-net. If $e^l \equiv e' \pmod{p}$ and e' is in the list as well, then $e = m$ with a high probability. As a result, the attackers know the message of the honest sender.

Clearly, those active attacks can be avoided by introducing redundancy into any message, more precisely, any mix-center must be able to check if the redundancy scheme is satisfied (Pfitzmann, 1994). This results into a special case of the original mix-net given by Chaum (1981). Otherwise, it seems hard to prevent the attack. In the second protocol it's possible to detect the attack (Pfitzmann, 1994) but in an electronic voting environment this is not sufficient.

As a result, it's possible to repair the weaknesses caused by the underlying insecure mix-net. It should be stressed that Pfitzmann does not guarantee the security of the modified mix-net and the voting scheme based on that modified mix-net. In fact, it will be shown in section 4 that the voting scheme is not secure, even if the mix-net is.

3 THE VOTING SCHEME BY PARK, ITOH AND KURUSAWA

The goal of the voting scheme by Park, Itoh and Kurusawa is to guarantee fairness. As the election phase can be disturbed by the mix-nets, it is assumed that the election will be repeated in this case. To avoid that intermediate results are leaked, the whole vote v is casted anonymously in encrypted shares. After being processed by the mix-net only some encrypted shares are decrypted in such a way that the whole vote v can't be recovered. Hence a voter can check if his shares are processed and decrypted correctly. If not, the election will be interrupted and repeated. Otherwise the rest of the encrypted shares are decrypted and the vote can be recovered.

Before describing the protocol in more detail, we review the underlying encryption scheme, which is an encryption scheme with multi-decryption.

3.1 An encryption scheme with multi-decryption

An *encryption scheme with multi-decryption* is a public-key cryptosystem in which one sender sends an encrypted message to n receivers. All n receivers are needed in order to decrypt the encrypted message, less than n receiver can't decrypt it.

Now we describe an encryption scheme with multi-decryption based on the ElGamal encryption scheme (ElGamal, 1985), as it will be used in the voting scheme later. This can be regarded as a special case of the ElGamal encryption scheme with threshold decryption due to Desmedt and Fraenkel (1989).

A trusted authority chooses a large prime p, an integer $q|(p-1)$, a generator $\alpha \in \mathbf{Z}_p^*$ and publishes them as system parameters. All receivers i choose a random number $x_i \in_R \mathbf{Z}_q^*$ and computes $y_i := \alpha^{x_i} \pmod p$. Each receiver publishes y_i and keeps x_i secret.

To encrypt a message $m \in \mathbf{Z}_{p-1}$, Alice picks a uniformly chosen random number $k \in_R \mathbf{Z}_q^*$ and computes

$$r := \alpha^{-k} \pmod p, K := (\prod_{i=1}^{n} y_i)^k \pmod p \text{ and } C := m \cdot K \pmod p.$$

Alice sends the tuple (C, r) to the receivers.
Every receiver computes $K_i' := r^{x_i} \pmod p$. Then all receivers can compute $K' := \prod_{i=1}^{n} K_i' \pmod p$ and $m := K' \cdot C \pmod p$.

3.2 Description of the voting protocol

The voting protocol consists of several phases. In the following we just focus on the vote of one voter. Clearly, many votes should be transformed together through the mix-net.

Initialization
Every mix-center generates a key pair for the encryption scheme with multi-decryption.

Registration
Every voter picks a key pair (x, y) for the signature scheme and transmits the public key through the mix-net. Mix-center n writes the received public-keys y on a list. They are called *pseudonyms* (Chaum, 1981), as there's no link between the voter and this public-key.

Casting of votes
If the voter wants to caste the vote v, he computes the tuples

$$(k_{1,1}, k_{1,2}), \ldots, (k_{l,1}, k_{l,2})$$

with $k_{i,1} \oplus k_{i,2} = v$ for $0 \leq i \leq l$. He encrypts each $k_{i,j} \circ 0^t$ (and the signature $s_{i,j}$ signed with the secret key related to the pseudonym) with the encryption scheme with multi-decryption, that is

$$c_{i,j} := E(y, s_{i,j}, k_{i,j} \circ 0^t)$$

and processes all values

$$(c_{1,1}, c_{1,2}), \ldots, (c_{l,1}, c_{l,2})$$

through the mix-net, so that all $c_{i,j}$ with $1 \leq i \leq l$ and $1 \leq j \leq 2$ are shuffled *together*.

Decrypting phase
The mix-net center n receives the values $c_{i,j}$ and for all i $(0 \leq i \leq l)$ and *exactly* one j $(1 \leq j \leq 2)$. Then all mix-centers decrypt together and compute

$$y', s'_{i,j}, k'_{i,j} := D(c_{i,j}).$$

Claiming phase
If $k'_{i,j}$ satisfies the redundancy scheme (the t lowest bits should by equal to zero), the messages are processed correctly. If this check fails, the voter claims this fact and the election will be repeated.

Decryption and Counting phase
If no voter claims that his opened shares are processed wrongly, the rest of his encrypted shares $c_{i,j}$ are decrypted as well. Then the vote v can be recovered, if at least one correct $k_{i,j}$ is obtained.

4 CRYPTANALYSIS

Now we show that independent of the used mix-net protocol this voting scheme can guarantee neither robustness nor fairness.

4.1 Robustness

It's possible that one mix-center deviates from the mix-net protocol. This will be detected in the claiming phase. However, it's not possible to detect *which of the mix-center is*

dishonest. Therefore, this disruption can be repeated ad infinitum. As a result, the protocol can be disrupted by any mix-net.

On the other hand, a dishonest voter can encrypt an arbitrary number which does not satisfy the redundancy scheme. Later, in the claiming phase, he can accuse the mix-centers wrongly that they have proceeded his vote falsely. Although the mix-centers will reject that, it can't be decided (by a referee) if the voter is honest and at least one of the mix-centers is dishonest or if the voter is dishonest and all mix-centers are honest. As a result, the protocol can be disrupted by any voter as well. A similar problem, but in another context, was discussed in Pfitzmann (1994) in section 5.2.

4.2 Fairness

It's further possible to show that fairness can't be guaranteed, although this was the main improvement in comparison to Chaum's voting scheme (1981) according to Park, Itoh and Kurusawa (1993).

Assume that no voter claims that his encrypted shares are processed falsely through the mix-net. Therefore, there's only a small probability of 2^{-l} that *all* encrypted shares which are not decrypted in the first decryption phase are processed wrongly. However, the conclusion that with an overwhelming probability the casted vote can be recovered in this case is incorrect: Assume that one mix-center is dishonest, but all mix-centers transforms the encrypted shares correctly. If all voters are honest, no one will claim that his vote is transformed wrongly. *However, the dishonest mix-center can deviate from the protocol in the decryption phase*, if the protocol described in section 3.1 is used, as done by Park, Itoh and Kurusawa (1993). Instead of computing $r^{x_i} \pmod p$ the dishonest mix-center i just broadcasts a random value K_i' to the other mix-centers. Clearly, as $k_{i,j}'$ does not satisfy the redundancy scheme, this attack will be detected and the election must be repeated. However, the dishonest mix-net can compute the vote in private and can reveal this knowledge. As a result, fairness can't be guaranteed.

Even more seriously, if the dishonest mix-center i knows the values $K_j' := r^{x_j} \pmod p$ of the other honest mix-centers it can determine the value K_i' suitably by computing

$$K_i' := \frac{\tilde{v}}{C \cdot \prod_{j=1, i \neq j}^{n} K_j'} \pmod p$$

and thus the vote $\tilde{v}$ will be computed as the recovered vote instead of the vote v which was casted by the voter. Therefore, the result of the election is wrong. Note that the attack can be detected only by the voter, but it's subtle for the voter to prove that a mix-center behaved dishonestly.

5 COUNTERMEASURES

A countermeasure to obtain robustness regarding dishonest mix-centers is to use a concrete mix-net protocol in which the correctness is proved. In this protocol no mix-center can't deviate from the protocol without being caught. However, it seems to be difficult to design such a protocol which is not vulnerable to active attacks as the one given by Sako and Kilian (1995). Clearly, once such a protocol is available the share-of-vote philosophy is

superfluous, that means, the whole vote can be transformed. Then the claiming phase can be eliminated as well. Thus the disruption by dishonest voters is also countermeasured.

On the other hand, it's possible to guarantee fairness by modifying the encryption scheme with multidecryption: In the decryption phase every mix-center proves the correctness of his calculations. The scheme in section 3.1 can be extended as follows: Every receiver i can show to anyone that he has decrypted correctly by proving in zero-knowledge that the discrete logarithm of K_i' to base r is the same as the discrete logarithm of y_i to base α using Chaum's proof (Chaum, 1990), if the ciphertext is authentic. We assume that a receiver is guilty, if he refuses to give the proof. Then any receiver, who cheats during decryption, will be caught. This technique was also used in the voting protocol by Horster, Michels and Petersen (1995). Alternatively, the proof DECRYPT in Sako and Kilian (1995) can be applied. Although this proof is less efficient it's possible to modify it into a non-interactive proof. As a result, it's impossible for a mix-center to cheat during decryption without being caught.

6 REFERENCES

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

Chaum, D. (1981) Untraceable electronic mail return addresses and digital pseudonyms. *Communcations of the ACM*, Vol. **24** (2), Feb., 84–8.

Chaum, D. (1990) Zero-knowledge undeniable signatures. LNCS 473, *Advances in Cryptology: Proc. Eurocrypt '90,* Springer Verlag, 458–64.

Desmedt, Y. and Fraenkel, Y. (1989) Threshold cryptosystems. LNCS 435, *Advances in Cryptology: Proc. Crypto'89*, Springer Verlag, 307–15.

ElGamal, T. (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. **IT-30**, No. 4, July, 469–72.

Fujioka, A., Okamoto, T. and Ohta, K. (1992) A practical secret voting scheme for large scale elections. LNCS 718, *Advances in Cryptology: Proc. Auscrypt'92*, Springer Verlag, 244–51.

Horster, P. and Michels, M. (1995) Der Vertrauensaspekt in elektronischen Wahlen. *Proc. Trust Center*, Vieweg Verlag, 180–9.

Horster, P., Michels, M. and Petersen, H. (1995) Blind multisignatures and their relevance to electronic voting. *Proc. 11th Annual Security Applications Conference*, IEEE Press, 149–55.

Park, C., Itoh, K. and Kurosawa, K. (1993) All/Nothing Election Scheme and Anonymous Channel. LNCS 765, *Advances in Cryptology: Proc. Eurocrypt'93*, Springer Verlag, 248–59.

Pfitzmann, B. (1994) Breaking an efficient anonymous channel. LNCS 950, *Advances in Cryptology: Proc. Eurocrypt'94*, Springer Verlag, 332–40.

Sako, K. and Kilian, J. (1995) Receipt-Free Mix-Type voting scheme. LNCS 921, *Advances in Cryptology: Proc. Eurocrypt'95*, Springer Verlag, 393–403.