E-Voting: An Electronic Ballot Box

Fabian Mauchle
Software and Systems
University of Applied Sciences Rapperswil, Switzerland
www.hsr.ch/mse

Abstract
Current non-electronic voting systems use a ballot box as one of their central element. Can the properties of such a classic ballot box be reproduced in an electronic voting system? Yes! A verifiable mixnet represents a method that can fulfil virtually any requirement a classic ballot box can.

1. Introduction
This document provides a description about a specific topic in e-voting, the ballot box, using a verifiable mixnet, the possible attacks and countermeasures. Additionally some comparison to the current voting system of Switzerland is provided. This document does not invent any new technology or method for e-voting, but is manly based on the work by Ben Adida (Adida & Rivest, 2006), (Adida, 2008). The main purpose is to help interested people in understanding the topic of e-voting.

Some of the very early voting systems in Switzerland, called ‚Landsgemeinde', where each voter held up his hand or a dedicated piece of paper when called for a decision, had one big problem: all votes were public. Everyone could see everyone else vote. Despite this fact, coercion was never noted a problem. Even today, some canton's (states) still use this system. But it has also it's advances: because everything is public, the voting process can be observed by everyone, and so cheating is almost impossible.

As the communities grew, and finally as the country today known as Switzerland was formed and the universal suffrage was introduced, the Landsgemeinde became unpractical and were mostly replaced by the today known systems with a ballot box. It developed even further such that in most parts of Switzerland, voting by mail became the usual case and has not to be applied for anymore. Compared to the early systems, the votes are now secret as they are put in a separate envelope. Despite this, coercion would still be possible at the time of filling in the ballot at home, but has since never led to large cases of cheating. On the other hand, the new system also raised new issues. Most of the voters will never have a look at the ballot box, and there is absolutely no guarantee that the mail with the votes reaches the voting office. To treat the first issue, the ballot box as well as the counting of the votes is usually observed by representatives of the different parties.

In the following sections, we will have a detailed look at the ballot box, and it's electronic replacement. The systems surrounding it like the handing in of the votes, or the counting of the votes will not be covered in detail. I necessary, the specific requirements will be explained.

2. The Ballot Box

As mentioned above, one of the major parts of a today's voting system is the ballot box. In this section, the purpose and properties of a ballot box and its electronic replacement - implemented as a verifiable mixnet - are discussed in detail.

2.1. The Classic Ballot Box

In today's voting systems, the ballot box is the core part where everything meets. The votes, written on a ballot, are dropped in the ballot box and thereby separated from the voter. After all ballots have been collected, the ballot box is opened and the votes are tallied. With this procedure, two main requirements of voting are ensured:

1. **Election secrecy**. To allow electoral freedom, the choice every voter has made, must be kept secret. Because the ballot must be read in order to be counted, there must not be any information about which person handed in the vote. To achieve this, all ballots look exactly the same (except the decision written on it) and / or are put in a uniformly looking envelope. In other words, the ballot does not contain information about the voter, nor does it hold any property which would allow an observer to attribute a vote to its voter.
2. **Immutability**. After a vote has been cast, it must be assured, that a vote cannot be altered and that it represents the will of the voter. This includes removing and altering as well as adding illegal votes. For that purpose, a ballot box has usually only a small slot to cast the ballot, and is sealed, such that ballots cannot be removed from the ballot box without somebody noticing. Thus, removing and altering a vote is inhibited. To ensure that no additional votes are dropped into the ballot box, it is closely observed while open to the public for passing in their votes and completely sealed after the voting deadline. Additionally every voter can check the integrity of the seal (see note below). The opening (and thus breaking the seal) for tallying is again closely observed to inhibit cheating.

The ballot box itself does not ensure some other important requirements of voting such as the inhibition of multiple votes by the same person or the cheating of the tallying process. These topics are outside the scope of this document and will not be discussed further.

Note: Because the voting in Switzerland is mostly done by mail, the properties of the ballot box are somewhat undermined. While the ballot box itself still fulfils the requirements, the security of the ballot is not fully guaranteed while it's carried by the post office. The ballot must be sent in a sealed envelope, but is not tied to the voter identification card. Thus the envelope could be replaced by one containing a desired vote, or dropped completely. Surely, a massive manipulation would be noticed at the voting office, but one has to basically trust the post office.

2.2. The Electronic Ballot Box

For the electronic variant of the ballot box, things are a bit different. While the requirements remain the same, the system is divided into two parts: one for the casting of votes, usually called the bulletin board, and one for the anonymization process. The focus of this document lies mainly on the anonymization (mixing) process, not on the bulletin board. Technical details about the bulletin board are not provided.

The bulletin board is responsible for collecting the ballots. It has to ensure, that every voter only has one vote, and that the casted votes cannot be altered. Usually, the votes are encrypted by a public key system, and therefore all votes are secret at this time. But the ballots are not yet separated from its voter. Because of that, there is also no security issue by observing the casting of votes. Other features may include a system for the voter to check that their votes have been correctly cast. Further requirements for the bulletin board are necessary to treat some security issues (see section 3).

In contrast to a classical ballot on a piece of paper, an electronic one is never uniform. The encrypted vote has its very unique bit pattern which could always be attributed to the voter. Therefore, additional anonymization procedures other than just mixing the sequence of the ballots are needed. Inspired by the classical mixnet anonymization networks, new systems have been developed (Park, Itoh, & Kurosawa, 1994). The special property in voting that all ballots can be collected first and mixed as a block afterwards, is very advantageous. It avoids the creation of dummy traffic and removes any relation about early-in early-out patterns as all ballots enter and exit the system at the same time. But the unique bit pattern still remains. To overcome this, the ballots must be altered – without altering its content – in some way. The classical approach was to encrypt the message (the ballot in case of voting) several times with different keys, while each mixing server decrypted one shell and passed the content to the next server. It is therefore called an onion router. The new approach makes use of the El Gamal reencryption .

In the El Gamal public key system (ElGamal encryption, 2009), there are defined: (All calculations are done in the finite field over p (all calculations $mod\ p$))

- The public factors: g, p (such that p is prime and g is a generator of a finite field G of order p)
- The private key: x
- The public key: $y = g^x$

To encrypt the message m, a private, random factor r is chosen. The cipher text c is calculated as

$$c = (\alpha; \beta) = (g^r;\ m * y^r)$$

And to decrypt this message

$$m = \frac{\beta}{\alpha^x} = \frac{m * y^r}{g^{r*x}} = \frac{m * y^r}{y^r}$$

For the use in the mixnet, the El Gamal crypto system defines a special function: the reencryption. Its special property is that it can alter the encrypted message, without altering its clear text content, and it only requires the public key, not the private key. To do this, an additional random factor r' is chosen. Then the new cipher text c' is calculated as

$$c' = (\alpha'; \beta') = \left(\alpha * g^{r'}; \beta * y^{r'}\right) = \left(g^r * g^{r'}; m * y^r * y^{r'}\right) = (g^{r+r'}; m * y^{r+r'})$$

The decryption remains the same

$$m = \frac{\beta'}{\alpha'^{x}} = \frac{m * y^{r+r'}}{g^{(r+r')*x}} = \frac{m * y^{r+r'}}{y^{r+r'}}$$

Using this reencryption function, a mix server can easily alter the bit pattern of a message without altering its content. Since the mix server only requires the public key, it can be fully independent from the key generation. Also the number of consecutive mix servers is unlimited.

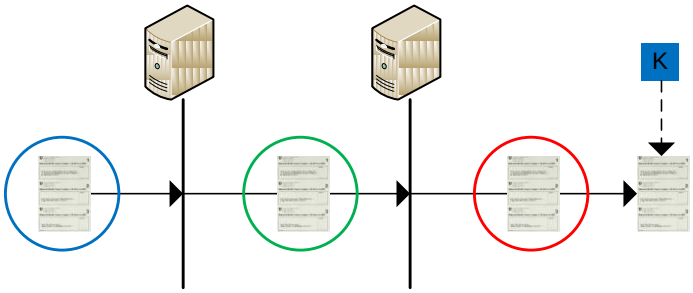

Figure 1: a reencryption mixnet

Assembled all together, the electronic ballot box consists of the bulletin board and one or more mix servers. After all votes have been cast, the complete list of encrypted ballots is passed to the first mix server, which randomizes (permutation π) the sequence of the ballots and does a reencryption with randomly chosen r' on each ballot. The resulting list is then returned to the bulletin board and passed to the next mix server, if applicable. In the end, the final resulting list of ballots – but only the final list – is decrypted and the votes are tallied. Decrypting any intermediate list might reveal the person who cast the vote and thus violates the election secrecy.

2.3. Trusting the Electronic Ballot Box

As the basic requirements for a ballot box (see 2.1), the question whether to trust the electronic ballot box can be divided into the same parts regarding the election secrecy and the immutability.

The first question about the election secrecy is more a political then a technical one. While a mix server reliably removes the attribution of the voter for the public, it could itself keep a list of the mappings. To solve this technically, several consecutive mix servers, each controlled by a different party, are used. If at least one mix server keeps its mixing secret, the whole process is secret. The question arising is whether to trust one single mix server (e.g. run by the government) not to reveal the mixing. As already mentioned, this question is more a political one, which cannot be answered in general. For Switzerland it might be ‘yes’, as similar trust is put in the post office and the municipal administration while transporting and collecting the ballots sent by mail.

Another part of the question about election secrecy is the generation of the private key. In the classical mixnets, each mix server has its own private key, or has contributed its part to the private key in case of El Gamal. But as the private key is not necessary for the mixing process, the question about who generates and owns the private key can be separated from the question

about the number of mix servers. As already mentioned in the last section, everyone in possession of the private key might decrypt an intermediate ballot list (or the first, non-mixed list) and compromise election secrecy. Therefore it is highly recommended to use a threshold cryptosystem (Threshold cryptosystem, 2009). Again, setting the right threshold is not a technical, but a political question.

The second question about the immutability mainly rises from the electronic ballot box being a 'black box'. This means that because it is an electronic system, it cannot be observed while working – and it must not, as this might again compromise the election secrecy - and the mix server could easily exchange, delete or add ballots at will. Thus, to be trustworthy, a mix server must prove in zero knowledge (meaning not to reveal the mixing data) that it did not cheat in some way. For this, the concept of shadow mixes is used:

A verifier can interactively challenge the mix server to test whether it cheated. This process can start immediately after the primary mix which will be tallied later, has been created. It can also be done at any later time as long as the mix server is available. At first, the verifier contacts the mix server with the request to challenge it. Upon this, the mix server creates a secondary shadow mix, which is exactly the same as the primary mix (the same input), but with different randomization factors t' and different permutation λ. The shadow mix is then returned to the verifier.

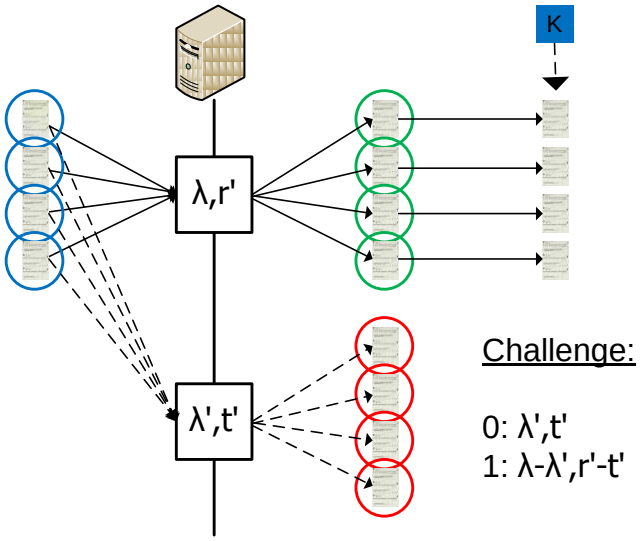

Figure 2: a secondary shadow mix

The verifier now challenges this secondary mix. He demands the mix server to reveal either information (chosen by the verifier):

- The randomization factors t' and permutation λ' used to create the shadow mix, or
- The difference between the primary mix and the shadow mix, that is, a permutation that recreates the primary shuffle out of the secondary ($\lambda - \lambda'$), and the corresponding randomization factors ($r' - t'$) needed to reencrypt the secondary mix in order to get the primary mix.

To prove correctness, the verifier recomputes the shadow mix out of the original input, or computes the primary mixing result out of the shadow mix and compares the result with the corresponding result of the mix server.

A mix server trying to cheat by replacing a ballot by its own, would have to decide, if it presents a correct secondary mix to the verifier, or a also cheated one such that the original, cheated mix can be computed out of the secondary mix. It is not possible to get both attributes correct with a cheated result. A cheating mix server has thus a 50% change of not being caught.

In order to get a better soundness, the verifier repeats the challenge several times. If only one challenge fails, the mix server is proven to cheat. In the Helios voting system (Adida, 2008), Ben Adida proposes to do 80 challenges, which leads to a soundness of 99.999999999999999999992%.

2.4. Speeding Up the Electronic Ballot Box

The interactive prove described above, puts a very high workload on the mix server, especially when multiple verifiers are challenging at the same time. It also requires high network bandwidth. Therefore, the proving of correctness may take very long. Besides the time needed, this also raises the risk that the mix server crashes before all verifiers are satisfied.

Amos Fiat and Adi Shamir introduced a scheme – sometimes referred as the Fiat-Shamir heuristic – to convert an interactive mechanism to a non interactive one. It mainly replaces the interactive challenge by a random oracle, which takes the data being challenged as input. As the output of the random oracle cannot be predicted (but is always the same for equal inputs) it can be used as the challenge. That way, the system to be proved can autonomously create the proof with no possibility to cheat. Please note that the random oracle is a theoretic entity which does not exist in reality. When implementing such a system, some sort of hash algorithm is used in place of the random oracle.

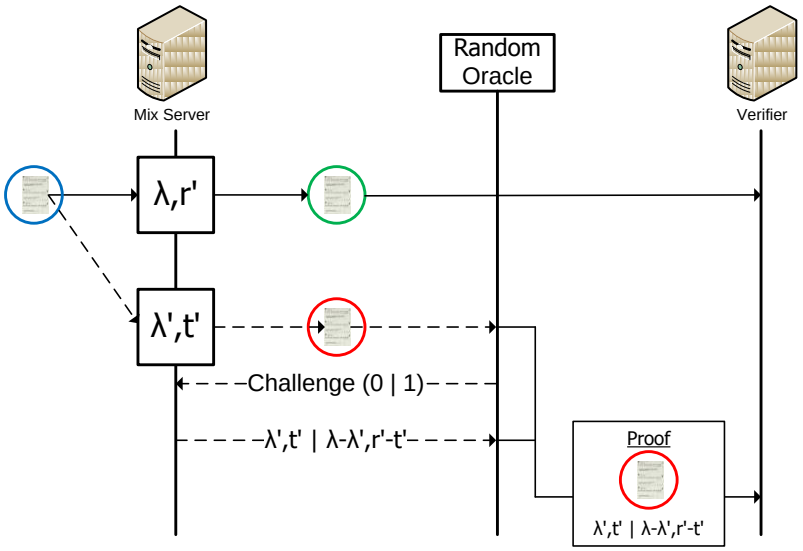

Figure 3: Fiat-Shamir heuristic for reencryption mixnet

This scheme can be applied to the shadow mixing prove described earlier. In that case, the mix server $1 + n$ mixes ($n = 80$ as proposed by Ben Adida). The primary mix being tallied at the end plus n additional shadow mixes for the challenge. It then concatenates all shadow mixes and uses this as the input to a hash function creating a hash of at least n bit. Each of these bits is used to challenge one shadow mix with the scheme described in 2.3. The mix server finally reveals the appropriate information according to the challenge bits, and publishes this information together with the n shadow mixes. Everyone wishing to check the prove can now recalculate the challenge and see if it is correct. Cooperation of the mix server is no longer needed. The security of the Fiat Shamir scheme relies mainly on the security of the used hash function.

In (Adida & Rivest, 2006) Ben Adida presents other variants of efficient prove which mainly reduce calculation effort needed. The Fiat-Shamir heuristic an also be applied to these variants. But as Ben Adida mentions later in (Adida, 2008), the time needed for calculation is not the critical factor. Amongst others, this is because most calculations can be parallelized very easily. Such variants will not be discussed further in this document.

3. Attacks and Countermeasures

As the properties of a ballot box can be divided into election secrecy and immutability, attacks on the electronic ballot box target either one. Since the mix server must provide a proof for correct mixing, cheating is very unlikely. Most attacks therefore aim to relate the output to the input and thus compromise the election secrecy.

3.1. Attacks to Election Secrecy

In (Adida & Rivest, 2006), Ben Adida lists two attacks against the election secrecy for the basic mixnet approach described in this document. The first is related to the security of El Gamal. (Pfitzmann, 1995) showed that the reencryption still leaves some mathematical relation, which allows pairing the inputs and outputs of the mix server. This is because El Gamal is not semantically secure in the usual case.

Pfitzmann also proposed a countermeasure to this attack, which was later proved to implement semantic security to El Gamal. The approach is on the generation of the keys, respectively its factors. Instead of just randomly choosing the large prime p and the generator g of $GF(p)$, p is generated as a safe prime, where another large prime q divides $p - 1$. The generator g is then selected for a q-order subgroup of $GF(p)$. Additionally, all plaintexts should also be in this q-order subgroup. This implies all cipher texts to be in the subgroup too.

The second attack is called the ‘related input attack’. The attacker selects the encrypted ballot he wants to track, and generates a mathematically related cipher text, which he casts has his ‘ballot’ – when decrypted, this ballot will not contain any useful or valid information, but only garbage. For the ballot $(\alpha; \beta)$ to be tracked, he chooses a random e and calculates $(\alpha^e; \beta^e)$ as his ballot. The reencyption done during the mixing preserves this mathematical relation, and, after the decryption, the output can be searched for the plaintext pair ${m_0}^e = m_1$. As one plaintext is only garbage, the other is the ballot which was tracked.

As a countermeasure, all voters are required to provide a non-interactive proof of knowledge of plaintext along with the encrypted ballot when casting it to the bulletin board. An attacker trying

to cast a ballot created by mathematically relating it to an already casted ballot will not be able to provide such a proof as he does neither know the original plaintext nor the resulting, mathematically related plaintext of his ballot. Details about how to provide a proof of knowledge of plaintext are out of scope of this document and will not be discussed further.

3.2. Cheating

As already mentioned at the beginning of this section, the proof of correct mixing required from the mix server makes cheating virtually impossible, as long as the number challenges is high enough. Although, when using the Fiat-Shamir heuristic, there might exist further security risks.

In (Goldwasser & Taumann, 2003) it is pointed out that applying the Fiat-Shamir heuristic opens a possibility for a forger to generate a challenge which will be accepted by the verifier. Applied to the mixing proof, this means that a mix server can cheat while still pretending to do a correct mixing. The background of this attack is that while a mix server never knows the challenge of the verifier, and is only very unlikely able to guess it, the challenge bits provided by a hash function can be calculated in advance. With this, the mix server is able to generate shadow mix – challenge combination that will prove correct thought it is cheated.

This attack is only limited by the calculation resources available to the mix server and an eventual time constraint the mix server has to provide the proof. It remains open, if increasing the number of shadow mixes to be provided also increases the effort for a mix server to find cheated shadow mix proving correct. Or if there exists an ideal number, after which the shadow mixes provide too much degrees of freedom, thus reducing the needed effort to cheat.

4. Conclusion

A verifiable mixnet is valuable component in an electronic voting system. It fulfils virtually any requirement a classic ballot box does. Implemented the right way, and with some external requirements also fulfilled, it can be considered secure today.

Beyond that, a verifiable mixnet adds some unique values a non-electronic voting system can never have. In the latter the ballot box is only observed by a few party representatives or other election observer. In an electronic voting system, the component that fulfils the function of the ballot box – the verifiable mixnet – can be observed by everyone who is interested in doing so. Thus the security is a lot higher than in non-electronic voting systems.

Bibliography

Adida, B. (2008). Helios: web-based open-audit voting. *Proceedings of the 17th conference on Security symposium* (pp. 335-348). San Jose, CA: USENIX Association.

Adida, B., & Rivest, R. L. (2006). *Advances in cryptographic voting systems.* Cambride, MA: Massachusetts Institute of Technology.

Bellare, M., & Rogaway, P. (1995). Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. (pp. 62--73). ACM Press.

ElGamal encryption. (2009, June 16). *Wikipedia, The Free Encyclopedia* . Retrieved November 9, 2009, from http://en.wikipedia.org/w/index.php?title=ElGamal_encryption&oldid=296679019

Goldwasser, S., & Taumann, Y. (2003). On the (In)security of the Fiat-Shamir Paradigm. *In Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science* (pp. 102-115). IEEE Computer Society Press.

Park, C., Itoh, K., & Kurosawa, K. (1994). Efficient anonymous channel and all/nothing election scheme. In T. Helleseth (Ed.), *EUROCRYPT. volume 765 of Lecture Notes in Computer Science*, pp. 248-259. Springer.

Pfitzmann, B. (1995). Breaking efficient anonymous channel. In A. De Santis (Ed.), *EUROCRYPT '94. volume 950 of Lecture Notes in Computer Science*, pp. 332–340. Springer.

Threshold cryptosystem. (2009, July 27). *Wikipedia, The Free Encyclopedia* . Retrieved November 9, 2009, from http://en.wikipedia.org/w/index.php?title=Threshold_cryptosystem&oldid=304440853