Verifiability in Electronic Voting
Explanations for Non Security Experts

Rojan Gharadaghy, Melanie Volkamer

CASED—Center for Advanced Security Research Darmstadt
Technische Universität Darmstadt
Mornewegstraße 32, 64293 Darmstadt
Germany
{rojan.gharadaghy, melanie.volkamer}@cased.de

Abstract: Scientists have requested verifiable electronic voting schemes for many years. These schemes offer individual and universal verifiability by applying and combining complex cryptographic primitives and protocols. Electronic voting systems in use provide less or even no verifiability. Thus election authorities and voters need to trust the provider and developer of the voting system regarding the integrity of the election. Due to arising critiques and the voting computer decision of the Federal Constitutional Court in Germany, the future electronic voting systems will probably need to implement verifiability. Therefore, this paper presents an overview and analysis of approaches to implement verifiability. We mainly address non-security experts like the average election authority and the average voter. Thus, the paper supports election authorities in their decision making process for a verifiable electronic voting system and the voter in making use of the verifiability.

1 Introduction

Electronic voting and in particular remote electronic voting offers many advantages compared to traditional paper based elections: like lower costs, faster tallying, improved accessibility and flexibility to the voter, greater accuracy of the result, lesser unintended invalid votes, and lower risk of human errors. However, an election can only profit from these advantages if the electronic voting system used ensures the four election principles of an equal, universal, secret, and free election. From an IT security point of view, these principles mainly mean that an electronic voting system has to ensure the secrecy of the vote and the integrity of the election result. Both must not be vulnerable to an outside attacker (i.e., hackers) nor to an inside attacker (i.e., developers, server/system hosts and administrators, and voters). Electronic voting systems used so far (e.g., in Estonia, the Netherlands, Austria, and Germany) have been evaluated by security experts. These evaluations mainly intended to check the system's robustness against outside threats while the election authorities and the voters have to trust that the developer and provider of the electronic voting system are not corrupt nor do they violate the election principles.

The problem is that once these systems are installed and the election is started, it is very difficult, if not impossible, to check whether the evaluated system and only this system is running[1] (for the entire voting period). As malicious providers could effect the system in several ways, (e.g., change the voting software undetected, log additional data, implement backdoors, change entries in databases in order to modify the election result or break the secrecy of the vote) verifiability mechanisms are necessary to run secure and trustworthy electronic elections. With these mechanisms voters and the public are able to audit the integrity of the election and thus the election result is ensured. Thus one can trust the provider, but the trust can also be audited. Note, even in traditional elections, trust in the people running the election (e.g., poll workers in the polling station) is not unlimited because observation in polling stations and other relevant places (e.g., central tallying) are allowed–sometimes required (compare [BWahlG, NRWO]).

Due to the fact that (a) the trustworthiness of a system rises if the system implements verifiability in addition to a security evaluation [Vo09]; (b) critiques have increased against the so-called black box voting systems; and (c) the German Federal Constitutional Court demanded verifiability for (electronic) voting in its voting computer decision [BFG09], the future electronic voting systems will probably need to implement verifiability mechanisms. The decision for a particular verifiability electronic voting system is made by the election authority, and the verifiability mechanisms themselves need to be applied by voters and observers. The problem is that verifiability approaches are based on complex cryptographic primitives and protocols. These approaches are only understandable to those with a background in cryptography. This is not the case for the average election authority or voter. Therefore, this paper presents an overview and analysis of existing technologies to implement verifiability from a non-security expert perspective. A couple of "important to know" statements have been identified and are labeled correspondingly. We point out the advantages and disadvantages of different approaches. The paper supports election authorities in their decision making process for a verifiable electronic voting system. It further helps voters to understand the advantages and boundaries of verifiable voting systems as well as to apply verifiability mechanisms in a future electronic election.

The remaining part of this paper is structured as follows: First, in Section 2, we give a short introduction on verifiability in general. The focus of Section 3 is individual verifiability and how this can be realized, while Section 4 concentrates on different aspects of and different approaches for universal verifiability. Section 5 concludes this paper.

[1] Trusted computing techniques could help here, but would require special hardware and software at the voter's PC. Therefore, these approaches cannot be applied to voting.

2 Verifiability

In the electronic voting literature, verifiability addresses the security requirement of the integrity of the election result. First of all, this means that it is possible for the voter to audit that his/her vote has been properly created (in general encrypted), stored, and tallied (the so-called *individual verifiability*). Further, this means that everyone can audit the fact that only votes from eligible voters are stored in a ballot box, and that all stored votes are properly tallied (the so-called *universal verifiability*[2]) [La09]. Systems providing both forms are called End-to-End (E2E) verifiable [Be09].

> **(Important to know 1)** Even with a verifiable electronic voting system, it is still possible for malicious providers and system developers to manipulate the (integrity of the) election result, but due to the verifiability, it will be detected. Thus it is not necessary anymore to trust them (regarding the integrity of the election result[3]).
>
> **(Important to know 2)** It is not required that each voter makes use of the individual verifiability or that all voters, candidates, parties, and observers make use of the universal verifiability. As a malicious provider does not know who verifies his/her vote and who does not, the provider cannot manipulate single votes without being detected with a very high probability. Regarding universal verifiability, it would even be sufficient if one trustworthy entity verifies the tallying.

Implementing verifiability in general would be easy. For instance a doodle[4] poll is perfect verifiable as everyone can go to the doodle web page after having cast a vote and verify that the vote next to his/her name has not been altered. Further he/she can verify that the result is correct by tallying each vote next to the voters' names (if the corresponding person is eligible to vote). But, if an electronic voting system needs to ensure the secrecy of the vote (which a doodle poll does not), it is necessary to apply and combine complex cryptographic primitives and protocols[5].

> **(Important to know 3)** Even with these cryptographic techniques, it is not possible to provide unconditional[6] verifiability and unconditional secrecy of the vote at the same time**.** Protocols ensure either unconditional verifiability and computational[7] secrecy of the vote or vice versa (compare [Ad08]).

---

[2] Other terms are public auditable or open audit [La09].
[3] Ideally, an electronic voting system would also provide the possibility to verify that the secrecy of the vote and maybe also other requirements are ensured (see [Vo09]). This is not covered in this paper.
[4] http://www.doodle.com/
[5] For electronic voting devices, it is also possible to realize verifiability without cryptography by using voter verifiable paper audit trails, printed by the devices and stored in a traditional ballot box. As the authors do not see a real benefit in these systems, this is not further addressed here.
[6] Unconditional means perfectly verifiable; even very powerful attackers cannot violate the integrity of the election result without been detected.
[7] Computational means that it depends on the solvability of a mathematical problem, which is classified as hard to solve. However, very powerful attacks would be able to break it. In general these problems are hard to solve today, but this might change in future.

Bulletin Boards (BB) have been invented in order to implement verifiability in electronic voting (for both remote electronic voting and electronic voting devices). BBs are public broadcast channels like web pages in the Internet, which have special properties: Data is published only by authorized parties and, once published, cannot be deleted or modified anymore. Such a Bulletin Board can be accessed (with read access) by everyone for verifiability purposes—including the voter and the election authority. The Bulletin Board contains and displays at least a list of cast votes (in encrypted form) together with voters' IDs or pseudonyms, and a couple of proofs used for verifiability (see Figure 1). The concrete content depends on the implemented verifiability approach. With the help of the Bulletin Board, verifiability can be done from any place at any time over the Internet—independent of whether the vote casting took place at an electronic voting system or over the Internet. Thus verifiability becomes possible for everyone not only those observing the process in the polling station. This is a main advantage compared to traditional paper based elections.

(Important to know 4) Bulletin Boards are a necessary concept to implement verifiability in electronic voting.

Verifiability can be achieved either "by hand" (by those who understand the underlying cryptography and are able to program their own verifiability) or by verifiability tools provided by independent institutes (for average voters and election authorities to run the verifiability). These could be downloadable or accessible through corresponding web pages. Moreover, the voter could also ask institutes like a university to run the verifiability on his/her behalf.

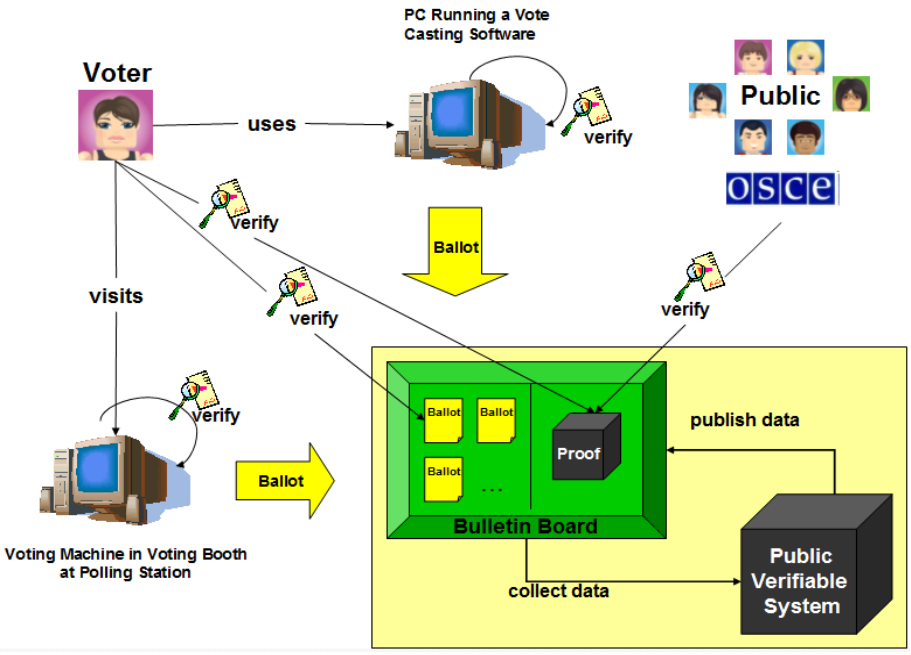

Figure 1: Overview on verifiability in electronic voting

(Important to know 5) Verifiability tools/software needs to be available from independent institutes so that the voter can choose the one he/she trusts.

2.1 Individual verifiability

Individual verifiability addresses the voter. The goal of individual verifiability is that the voter can verify that

(a) his/her vote is properly encrypted (during vote casting), i.e., if he/she chooses candidate *A* then candidate *A* is also encrypted and not candidate *B* [cast as intended];
(b) his/her encrypted vote is sent and stored unaltered at the Bulletin Board (after vote casting), i.e. if candidate *A* has been encrypted to $enc(vote_A)$ then $enc(vote_A)$ must appear on the Bulletin Board next to the voter's ID/pseudonym and not $enc(vote_B)$ [stored as cast];
(c) his/her encrypted vote is properly included in the election result (after tallying), i.e., in general properly decrypted and properly added to the other decrypted votes [tallied as stored].

Part (c) is only covered indirectly by universal verifiability. The idea is that if it is verifiable for all encrypted votes on the BB that they are properly included in the tally then this also holds for a particular vote [La09].

(Important to know 4) Individual verifiability ensures that the vote is cast as intended and stored (on the BB) as cast (part (a) and (b)).

(Important to know 5) Due to part (b), there exist a link between the voter/ pseudonym and his/her encrypted vote on the Bulletin Board. Consequently, once the encryption scheme is broken the secrecy of the vote is violated, if there is a link between voter and encrypted vote.

2.2 Main idea and first approach

Implementing individual verifiability can be realized relatively easy in the following way: votes are encrypted probabilistic, that is, votes are concatenated with a random number and then encrypted[8] *enc(vote#*[9]*R)*[10]. In general, knowing the values *vote* and *R* means that it is possible to "decrypt" this term without the knowledge of the decryption key: just by encrypting the value *vote#R* again and comparing the output with the encrypted term *enc(vote#R).* Based on this, the individual verifiability can be implemented in the following way:

[8] Public-key encryption is used, which means that a message is encrypted with the public key of the receiver, and the encrypted message can only be decrypted with the corresponding secret key, which is only known by the receiver.
[9] The symbol # is used for concatenations.
[10] Without this value *R* the encryption does not really protect the confidentiality of the vote as an attack could easily encrypts all possible votes and compares the output with *enc(vote).*

- The voter uses an individual verifiability tool.
- This tool gets as input from the voting application the encrypted vote *enc(vote#R)* and the random value *R* used for the encryption plus from the voter the value *vote*.
- This tool audits whether *the vote has been encrypted properly*.
- If this is the case, the encrypted vote is transferred to the Bulletin Board and stored.
- In order to later verify that the vote is properly stored on the BB, the random number is stored on the voter's PC.
- After having completed the vote casting, a voter can use the individual verifiability tool again to verify whether *his/her encrypted vote is properly stored* on the Bulletin Board.
- This time, the tool takes as input the stored random value *R* and from the voter, his/her choice and some personal information to identify the voter's entry on the BB.
- With this information the tool computes *enc(vote#R)* and verifies whether this value is on the Bulletin Board. Note, both verifiability checks can be repeated with arbitrary individual verifiability tools.

The described approach provides unconditional individual verifiability. But, it violates the secrecy of the vote because it is not receipt-free[11]. A voter could use his/her knowledge of the randomness *R* as a receipt to prove to himself/herself that he/she cast his/her vote.

2.3 Advanced approach

In order to avoid such a receipt and thus be receipt-free, the above described mechanism for individual verifiability needs to be modified in the following way (see, e.g., [Ad09, Ad08]):

- Here, after the voting application has encrypted the vote, the voter needs to decide whether he/she wants to verify that the *vote has been properly encrypted* or whether the voter wants to cast the vote (which means the encrypted vote is sent to and stored on the Bulletin Board while the encrypted vote is stored on the voter's PC[12]).
- Only, if the voter decides to verify the encrypted vote, the random value *R* is revealed as input for the individual verifiability tool.
- If the voter decides to cast the vote, the value *R* is not revealed to ensure receipt-freeness.
- In this approach, the *second part* of the individual verifiability works as follows: The voter uses the individual verifiability tool again.

[11] Receipt-free means that the voter does not get a receipt to prove which candidate he/she chose.
[12] In this approach, the randomness *R* is neither leaked to the voter nor stored on his/her PC.

- This tool takes as input the stored encrypted vote and from the voter some personal information to identify his/her entry on the Bulletin Board. It verifies whether the encrypted vote appears on the BB.

The consequences for the individual verifiability in this approach are the following:

- [cast as intended] The voter cannot verify whether the cast encrypted vote contains his/her candidate choice. After having successfully verified a couple of (test) votes, the voter has good evidence that his/her cast vote is also properly encrypted. The idea is that the voting application does not know how the voter will decide and thus does not know when (in case it would be malicious) to encrypt a different vote.
- [stored as cast] While in the previously described approach the voter could verify that the encrypted vote on the BB contains his/her candidate choice, in this approach he can only check whether the stored encrypted vote is properly stored on the Bulletin Board. However in combination with the evidence from the first part of the individual verifiability (cast/encrypted as intended), this is acceptable.

(Important to know 6) In order to verify that his/her vote is cast, the voter needs to verify his vote twice: once during vote casting and once after vote casting/after tallying. Thus there are two additional steps compared to black box voting systems if the voter wants to apply individual verifiability.

(Important to know 7) In order to provide receipt-freeness, a voter gets only evidence with high probability but no formal proof for the individual verifiability because the vote he/she finally cast cannot be verified, but only arbitrary votes before.

3 Universal verifiability

Universal verifiability is more complex than individual verifiability. At least two comparable (cryptographic) approaches exist. This section is structured into the following subsections: In the subsection 3.1, the main idea is proposed as well as its high level implementation and challenges in realizing it. The two main cryptographic approaches (one based on so called MIX networks and the other one based on the homomorphic property of encryption schemes) are introduced and explained in subsection 3.2.

3.1 Idea and Challenges

After the voting period for each voter who cast a vote, a corresponding encrypted vote is stored and published on the Bulletin Board[13].

13 Each encrypted vote can be unambiguously linked to a voter or his/her pseudonym. This is necessary to enable the individual verifiability.

(Important to know 8) The universal verifiability needs to ensure that all of these stored votes are properly tallied[14]. This usually also includes that the decryption is done properly.

The easiest way to implement universal verifiability would be to decrypt each vote on the Bulletin Board and publish all decrypted votes and the decryption/secret key. These data enable everyone to tally the votes him/herself or by using a universal verifiability tool and to verify that the votes are properly decrypted with the decryption/secret key. However, this would violate, in the worst case, the secrecy of the vote (if the encrypted votes are linked to the voters ID), and in any case, the system would not be receipt-free. Thus the Bulletin Board would contain either the information: *voter ID – encrypted vote – decrypted vote* or at least *voter's pseudonym – encrypted vote – decrypted vote.*
Obviously, it is challenging to compute the election result without violating the secrecy of the vote and being receipt-free. Therefore, one of the following two cryptographic techniques is applied to meet this challenge with corresponding cryptographic protocols (compare to [Sc08, Sm05])[15]:

- Cryptographers have developed encryptions schemes (so called homomorphic schemes), which allow the encrypted sum of all encrypted votes to be computed. Decrypting this sum is equal to the sum of all decrypted votes, i.e., $dec(enc(vote1) + enc(vote2) + \ldots + enc(voten)) = dec(enc(vote1)) + dec(enc(vote2)) + \ldots + dec(enc(voten))$. The main advantage of this approach is that it is not necessary to decrypt single votes. The decryption/secret key is only used once to decrypt the result. Therefore, the secrecy of the vote is ensured at the same time (see also Figure 2 and compare to Section 3.2.1).
- The second approach is based on the idea that the encrypted votes are first anonymized and then decrypted. To do so, the encrypted votes are first of all separated from the voter's ID/pseudonym. This set of encrypted votes is then anonymized by using a so called MIX net (compare to [Ch81]). A MIX net contains several nodes (so called MIX nodes which are general servers, running a particular software) while each MIX node takes as input the set of encrypted votes, shuffles this set and outputs a list of anonymized encrypted votes. This is done by each MIX node. Finally after shuffling the votes several times, the anonymized votes are decrypted and tallied. Several MIX nodes are used to increase the trust in the secrecy of the vote; although it is enough if one MIX node is trustworthy and anonymized the set of encrypted votes by shuffling the encrypted votes (see also Figure 3 and compare to Section 3.2.2).

[14] [Ry09] also recommends verifying that all votes are cast by eligible voters. We agree that this is necessary. However, due to time and space constraints this is not covered by universal verifiability in this paper.
[15] Actually in [Sm05], two more approaches are named. However, these are not very popular and thus not included in this paper. They are "heterodox schemes" and "schemes based on secret sharing among several mutually distrustful election authorities."

(Important to know 9) Two different approaches are distinguished for a tallying procedure that ensures the secrecy of the vote: (a) Either only the encrypted sum is decrypted (while single votes are never decrypted) or (b) the encrypted votes are anonymized by randomized shuffling and only the anonymized votes are decrypted.

A *universal* verifiability tallying procedure needs to ensure the secrecy of the vote while providing proofs of the election result's integrity, that is, proving that the tallying procedures ran appropriately. Thus proofs need to be created during the tallying. This makes the tallying more complex and also a little bit slower. Although it becomes more complex and involves more entities in the tallying, the robustness of the tallying needs to be ensured, i.e., running the tallying should not relay on single entities. No single (malicious) entity should be able to block the computation of the election result, e.g., by claiming to having lost the decryption/secret key.

(Important to know 10) A universal verifiability approach needs to ensure the secrecy of the vote and needs to be robust while providing proofs of the election result's integrity.

In order to get a universal verifiable voting scheme, it is necessary to extend the above described approaches (homomorphic encryption function / MIX nets) with corresponding proofs.

3.2 Two main approaches

The main idea of universal verifiability is to ensure that the tallying is done properly. Thus for both approaches, we explain what can go wrong in terms of where manipulations of the election results' integrity can occur and which techniques can be used to provide universal verifiability, i.e., make such manipulations detectable.

3.2.1 Approach based on homomorphic encryption

In a universal verifiably scheme based on homomorphic encryption, the following two manipulations must be detectable with corresponding proofs:

- The system provides an arbitrary result as output for the decryption of the encrypted sum.
- The key holder(s) get the wrong decryption/secret key. This wrong key is used for decryption. The corresponding output is not equal to the sum of decrypted votes. Thus the integrity of the election result is not ensured.

Further, the robustness of the tallying procedure should not depend on the one key holder of the decryption/secret key.

To improve the robustness, the secret key is shared by several authorities with a so-called secret sharing scheme [Sh79]. This can be done in a way that *k* out of *n* authorities are already able to decrypt a message (in this scheme the encrypted sum of all votes). Thus if some authorities lose their key shares, the result can still be determined. To overcome the problem of the key holders receiving the wrong keys, so called verifiable secret sharing schemes are applied [Ch85]. Here it can be proven that the shares are properly distributed. Cryptographers also developed methods to prove that a message was properly decrypted without revealing the secret key (this is necessary to ensure the secrecy of the vote). One possibility of proving the correctness of a decryption is to use the Chaum-Pedersen protocol [CP92]. Using all three techniques the tallying is universal verifiable and at the same time proofs are provided in two situations: one after the key distribution and the other one with the decryption of the election result. Correspondingly, in both situations the proofs need to be verified. Moreover, it needs to be verified that the encrypted sum has been calculated correctly. An overview of the universal verifiability approach based on homomorphic encryption is shown in Figure 2.

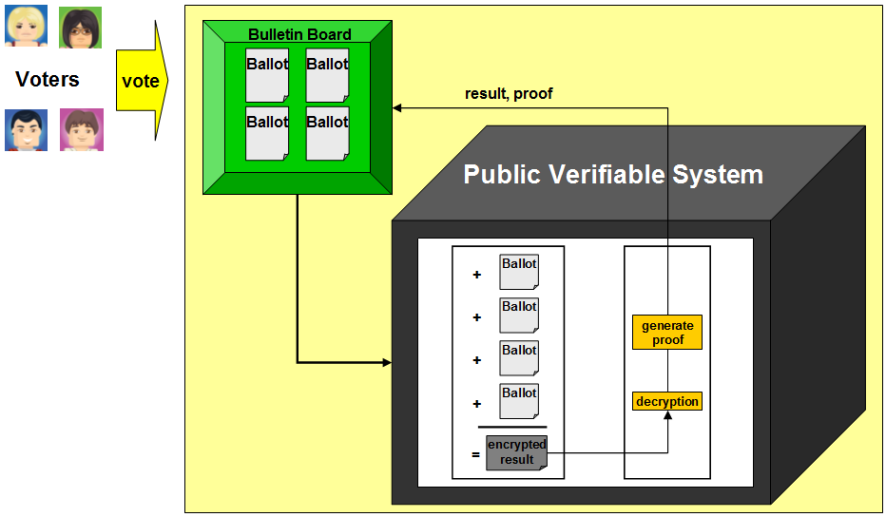

Figure 2: Universal verifiability based on homomorphic encryption

3.2.2 MIX-based approach

In a universal verifiably scheme based on homomorphic encryption, the following three manipulations must be detectable:

- The output of a MIX node does not correspond to the shuffled input because encrypted votes have been modified. (1)
- The component finally decrypting the votes provides an arbitrary result for the decryption of each vote. (2)

- The key holder(s) got the wrong decryption/secret key. This key is used to decrypt votes. The corresponding output is not equal to the cast vote. (3)

Further, the robustness of the tallying procedure should not depend on the one key holder of the decryption/secret key and not on each MIX node. To increase the robustness of the MIX net so called re-encryption MIX nets are used. This means that arbitrary MIX nodes can fail and arbitrary new MIX nodes can be added to increase the trust in the secrecy of the vote. To increase the robustness with respect to the key holder and to ensure (2) and (3), the same techniques as for the homomorphic approach are used (namely: verifiable secret sharing and a proof of correct decryption). In addition, it needs to be ensured that each MIX node cannot manipulate the election result by altering the votes from the input to the output. To do so, cryptographers either use Zero Knowledge Proofs or Randomized Partial Checking [JJR02]. The first provides a real proof while the second approach only provides high evidence. However, the second approach is much more efficient than the first one.

Using all these techniques, the tallying is universal verifiable, and proofs/evidences are provided in three situations: (similar to the homomorphic approach) one after the key distribution and the other one with the decryption of the election result; plus the proofs/evidence provided by each MIX node. Correspondingly, in all three situations the proofs/evidence needs to be verified. The universal verifiability approach based on MIX nets is shown in Figure 3.

(Important to know 11) While there is less effort involved in the tallying and verifiability of homomorphic schemes, not all election schemes can be run using this approach because for some schemes (e.g., with write-in ballots) a corresponding homomorphic encryption scheme does not exist.

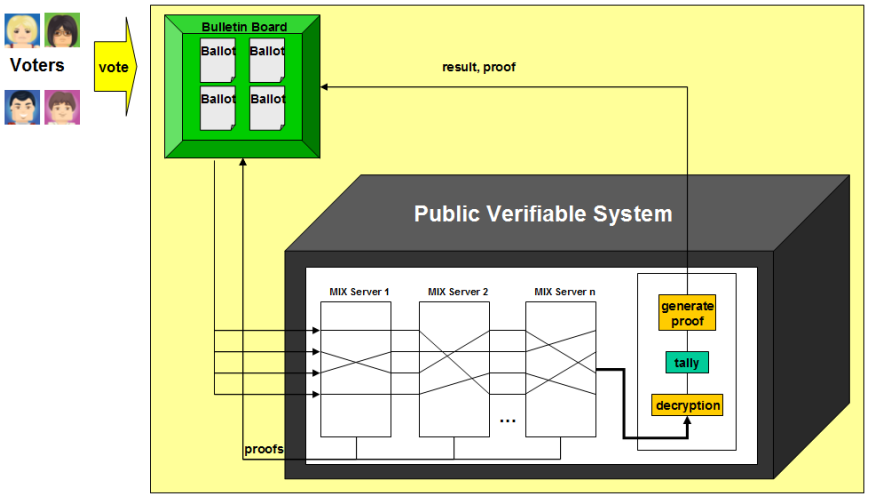

Figure 3: Universal verifiability based on a verifiable MIX net

4 Conclusion

Verifiability (both universal and individual) in electronic voting is becoming more and more important. After having discussed these techniques for years in the research community, this now needs to be implemented in future electronic voting schemes. Due to the fact that these techniques need to ensure the secrecy of the votes, the approaches are rather complicated and suffer from different constraints. The most important one is the theorem that an electronic voting system can either ensure unconditional secrecy or unconditional verifiability. Further, the election authority has to decide which verifiability approach they are in favor of.

This paper explains the different approaches from a high level perspective to also enable non-security experts to decide which technique to use and what its advantages and disadvantages are. Further, this paper addresses voters to help them understand what the advantages of verifiable voting schemes are and how to use them.

However, even if this paper helps to understand verifiability in the context of electronic voting, in order to use these techniques for legally binding elections, there are two open issues: First of all, the user friendliness has to be increased to enable average voters to use the verifiability mechanisms. Second, it is necessary to develop technical and/or organizational mechanisms and policies to handle those cases in which the result of any verifiability is negative.

References

[Ad08] Adida, B. 2008. Web-based open audit voting. In *Proceedings of the 17th symposium on security*, pp. 335–348. Berkeley, CA, USA: USENIX Association.

[Ad09] Adida, B. et al. 2009. Electing a niversity President Using Open-Audit voting: analysis of real-world use of Helios. In *EVT'09. Proceedings of electronic voting technology workshop*. USENIX Association.

[Be09] Benaloh, J. 2006. Simple verifiable elections. In *EVT'06. Proceedings of the USENIX/accurate electronic voting technology workshop*. Berkeley CA, USA: USENIX Association.

[Ch81] Chaum, D.. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. In *Communications of the ACM 24, February, Nr. 2*, pp. 84–90..

[CP92] Chaum, D. and T. P. Pedersen. 1992. Wallet databases with observers. In *CRYPTO, volume 740 of LNCS*, 89–105. Springer.

[Ch85] Chor, B. et al. 1985. Verifiable secret sharing and achieving simultaneity in the presence of faults. *In FOCS85*, pp. 383-395.

[BWahlG] *Bundeswahlgesetz in der Fassung der Bekanntmachung vom 23. July 1993 (BGBl. I S. 1288, 1594), last change 17. March 2008 (BGBl. I S. 394).*

[NRWO] *Bundesgesetz über die Wahl des Nationalrates (Nationalrats-Wahlordnung 1992 – NRWO) BGBl. Nr. 471 idF BGBl. I Nr. 28/2007.*

[BFG09] *BVerfG, 2 BvC 3/07 vom 3.3.2009, Absatz-Nr. (1–163), Urteil des Zweiten Senats*. http://www.bverfg.de/entscheidungen/cs20090303_2bvc000307.html/.

[JJR02] Jakobsson, M., A. Jueles, and R. L.Rivest. 2002. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th symposium on security*, pp. 339–353. Berkeley, CA, USA: USENIX Association.

[La09] Langer, L. et al. Unpublished. Towards a framework on the security requirements for electronic voting protocols. In *Post Proceedings of RE-Vote09*.

[Ry09] Ryan. M. 2009. *Verifying electronic voting protocols in the applied pi calculus*. Slides presented at the 3rd workshop on security and electronic voting (VETO 09). http://www-veto2009.imag.fr/Material/Mark Ryan.pdf/.

[Sc99] Schoenmakers, B. 1999. *A simple publicly verifiable secret sharing scheme and its application to electronic voting*. Technical report. Eindhoven, NL: Department of Mathematics and Computing Science, Eindhoven University of Technology.

[Sc08] Schoenmakers, B. 2008. Voting schemes. Draft book chapter. To appear in *Algorithms and theory of computation handbook, http://www.win.tue.nl/~berry/papers/ChVotingSchemesJuly2008.pdf*

[Sh79] Shamir, A. 1979. How to share a secret. *Communications of the ACM* 22 (11): 612–613.

[Sm05] Smith, W. D. 2005. Cryptography meets voting. http://www.math.temple.edu/~wds/homepage/cryptovot.pdf

[Vo09] Volkamer, M. et al. 2009. Elektronische Wahlen. Verifizierung vs. Zertifizierung. *Proceedings of INFORMATIK 2009, volume 154 of LNI*, 1827-1836. Bonn, Germany: Gesellschaft für Informatik.