Improving the verifiability of the Estonian Internet Voting scheme

Sven Heiberg[1,2], Tarvi Martens[2], Priit Vinkel[3], and Jan Willemson[4,5]

[1] Smartmatic-Cybernetica Centre of Excellence for Internet Voting, Ülikooli 2, Tartu, Estonia
[2] Electronic Voting Committee, Lossi plats 1a, Tallinn, Estonia
[3] Chancellery of Riigikogu, Secretariat of National Electoral Committee, Lossi plats 1a, Tallinn, Estonia
[4] Cybernetica, Ülikooli 2, Tartu, Estonia
[5] Software Technology and Applications Competence Centre, Ülikooli 2, Tartu, Estonia

Abstract. We describe an update of the Estonian Internet Voting scheme targeted towards adding verification capabilities to the central system. We propose measures to ensure the auditability of the correctness of vote decryption and i-ballot box integrity. The latter will be improved to a level where it would be possible to outsource the vote collection process to an untrusted party and later fully verify the correctness of its operations.

1 Introduction

In 2005, Estonia became the first country in the world to cast votes over the Internet for state-wide legally binding general elections. In the 2014 and 2015 elections, more than 30% [3] of all the votes were cast this way, making online voting the second most popular means of vote-casting after paper voting in polling stations on election Sunday.

The scheme used in 2015 was still more or less the same as designed for the first Internet-enabled elections in 2005. It mimics double-envelope postal voting, where the inner, privacy-providing envelope is replaced by encrypting the vote using the central system's public key, and the outer authenticity and integrity layer is provided by signing the vote cryptogram with the voter's ID card [6].

While it is straightforward to understand and sufficiently simple to actually implement in practice, the resulting system relies on several external assumptions. In the early days of implementation, the voter's computer was explicitly trusted. By 2011 it had become apparent that this assumption could not be relied upon any longer [6]. As a solution, the scheme was augmented with the option of individual verifiability using an independent mobile computing device [8].

However, individual verifiability alone is not sufficient to mitigate all the risks. For example, the Estonian system has recently been criticized by Springall *et al.* for its excessive reliance on physical and organisational measures [13].

Even though these measures have worked well in practice, auditing them is a non-trivial task that can only be performed by a limited set of trustees.

Even more importantly, both the process and outcome of such an audit are defined in a way that leaves a lot of room for human interpretation, and consequently also errors. The possibility of such errors can be used to raise doubts and these doubts can in turn be used against Internet voting in political debates.

The aim of this paper is to describe the second major update of the Estonian Internet Voting scheme that is targeted towards adding verification capabilities to the parts of the central system that have so far been the most difficult to audit. More precisely, we will propose and discuss measures to ensure third-party auditability of the correctness of vote decryption and i-ballot box integrity. The latter will be improved to a level where it, in principle, becomes possible to outsource the vote collection process to an untrusted party and later fully verify the correctness of its operations.

2 Estonian Internet Voting scheme

On the conceptual level, the Estonian Internet Voting scheme used in 2005–2015 mimics double envelope postal voting [6]. The core system consists of the Voting Application ($VoteApp$), the Vote Forwarding Server (VFS), the Vote Storage Server (VSS) and the Tabulation Application (TA) with the Hardware Security Module (HSM) for private key protection. The online components log to the Log Monitor (LOG), and there is a OCSP responder ($OCSP$) that provides both certificate validation and time-marking services.

The central voting system generates an RSA keypair with the HSM and publishes the public part ek^{elec}_{pub}. The voter v uses the $VoteApp$ and authenticates herself for the VFS using her smart-card-based digital identity tool, and receives the candidate list. She then makes her choice c_v and encrypts it with the systems's public key.

For encryption, RSA-OAEP is used and a random number r_v is generated for encryption. Hence the anonymous ballot ("inner envelope") is computed as $ballot_{c_v,r_v} = Enc(c_v, r_v, ek^{elec}_{pub})$. The effect of the "outer envelope" is achieved by signing the ballot using the voter's digital identity tool, and the resulting vote $vote_v = Sign(sk^v_{priv}, ballot_{c_v,r_v})$ is sent to the VSS for storage.

Electronic ballots are stored in the signed and encrypted form until the voting period is over. The signatures are then dropped in VSS and anonymous ballots are tallied in TA. For that, they are decrypted with the server's private key stored in a HSM.

In 2013, the scheme was augmented with the option of individual verifiability [8]. The randomness r_v and the unique vote identifier vid generated by the central system are made available by $VoteApp$ to a mobile device running a Verification Application ($VerApp$) in the form of a QR code. The identifier is used by the $VerApp$ to request the inner envelope $ballot_{c_v,r_v}$ from the VSS. The process then uses the list of candidates C and the randomness r_v to find a

$c' \in C$ such that

$$Enc(c', r_v, ek_{pub}^{elec}) = ballot_{c_v, r_v}.$$

It is up to the voter to decide if the outcome of this process was expected or not.

3 Shortcomings of the current scheme

The security analysis [1] of the Internet voting concept described in Section 2 identified security requirements broadly divided into the categories of integrity, confidentiality, transparency and coercion-resistance. A set of measures to mitigate the identified risks was provided under assumptions about the operating environment – the existence of a reliable PKI, the supremacy of paper-voting, the trustworthiness of the Internet voting system and its operations, and the trustworthiness of the voters' computers.

Some of these assumptions – the trustworthiness of the voters' computers – are not considered valid anymore. These new considerations have led to system improvements, e.g. the addition of individual verifiability [8]. Other assumptions – the existence of a reliable PKI – still hold in Estonia.

Assumptions about the trustworthiness of the central system lie somewhere in between. On the one hand, there are a number of physical and organizational measures to ensure them, but on the other hand, such measures can always be questioned [13]. The current nature of these measures increases the involvement of the National Electoral Committee in organizing online voting to the point where it needs to perform the technical tasks of hosting that could normally be outsourced to an external online service provider. Thus, the general goal of this paper is to redesign the Estonian Internet Voting system to become less dependent on the human factor, allow more independent verifiability and a better separation of duties between different organizations.

In the rest of this Section, we will review the main challenges of the current system that will need to be addressed. As a starting point of our analysis, we will use the attack tree presented by Heiberg and Willemson in [7]. We will exclude the availability-related attacks from the discussion and assume the existence of an individual verification tool to detect manipulations on the voters' computers.

3.1 I-ballot box integrity

Three direct attacks on the i-ballot box integrity can be identified: adding votes to the box, removing votes from the box, and modifying votes already in the box [7].

The process of vote storage takes advantage of the Estonian PKI with digital signature capabilities and private keys stored on secure hardware tokens.

- $VoteApp$ creates the digitally signed vote and sends it to the VFS.
- VFS verifies the signature of the vote and forwards the vote to the VSS.

– VSS verifies the signature and acquires confirmation about the validity of the certificate to the signature from $OCSP$.
– VFS and VSS log the stages of processing both locally and to the LOG.

PKI usage allows us to assume that the stored votes are secured from the manipulations and that the eligibility of voters can be verified by the system. Unauthorized addition or modification of the votes would effectively require forging digital signatures. Estonia has relied on its digital signature infrastructure since 2002, and we can argue that potential weaknesses leading to signature forgery have been mitigated.

However, there are no comparable measures against the unauthorized removal of votes from the i-ballot box. An attacker who wants to remove a vote from the system has to compromise the VSS in a way that it would be possible to delete the corresponding file from storage. The attacker must take the following risks into account:

– There is a certain window of time during which a vote can be verified by the individual verification tool. If the vote is removed before the end of this window, there is a risk of detection.
– There are traces of vote storage in the log files on VFS, VSS, LOG and $OCSP$. If these traces are not removed and the logs are later correlated with the actual list of stored votes, there is a risk of detection.

Potential detection by the individual verification tool can be easily prevented by deleting the vote after the verification window (30 or 60 minutes) has closed.

Tampering with the log files requires control – such as administrator access – over multiple components. The remaining risk for the attacker is that the $OCSP$ is hosted completely independently from the Internet voting system. Currently the consistency of the VSS and $OCSP$ views is not rigorously audited, which leaves the unauthorized removal of the votes from the i-ballot box a theoretical possibility.

3.2 Tabulation integrity

The following attacks on tabulation can be identified: i-ballot box replacement, tabulation tool compromise, and forgery of the voting result [7].

There are several steps in the process of tabulating votes.

– VSS verifies the signatures of the stored votes and extracts a set of encrypted votes sent to the tabulation.
– TA takes the set of encrypted votes and decrypts them with the private key stored in the HSM.
– TA aggregates and digitally signs the voting result.
– Both VSS and TA log the status of each encrypted vote. These logs are later audited.

We argue that the threat of actual forgery of the voting result has a valid countermeasure as there exists an audit procedure, which involves retabulating the votes and comparing the result with the published one.

An attacker wishing to replace the i-ballot box before tabulation would have to compromise the VSS. Although the VSS is offline in this stage of the election, the same system is online during the voting period. This makes a remote compromise possible as well. A malicious VSS would simply replace the set of encrypted votes sent to tabulation. It would also use the forged set of encrypted votes as basis for audit log forgery. There is a risk of detection for the attacker – it is possible to repeat the process of anonymization on the original set of signed votes using a different combination of hardware and software. Currently this kind of audit has not been implemented.

An attacker who has compromised the tabulation tool can take advantage of the fact that right now, there is no way to either verify or audit if the encrypted ballots are decrypted correctly. A flaw in the tabulation tool – TA together with HSM – could change the result without anyone noticing.

In the current scheme, the integrity of tabulation relies on the correctness of the software and hardware together with the integrity of the operating personnel. If we want to weaken the assumption that the central system is trustworthy and outsource aspects such as online vote collection to a third party, additional countermeasures are required for both i-ballot box replacement and tabulation tool compromise.

4 Towards the solution

It is possible to make statements about the integrity of the voting result; the question is how can we prove these statements in a non-disputable manner. Take the example of traditional paper-based systems that can be audited by a full or random-sample recount. Out of the two, a full recount has been established as the common ground for resolving such disputes. Unfortunately, a full recount of all paper ballots is resource-intensive and error-prone due to human inaccuracy in both marking and counting the ballots. Recent research by Goggin *et al.* shows that the margin of error of paper ballot counting can be reduced to about $1 \ldots 2\%$, but not much lower [5].

In the era of computer technology we can actually do better. The corresponding solutions are generally known as providing *end-to-end verifiability*, i.e. allowing to check that certain properties regarding the relationship of the stored ballots and the voting result actually hold.

In our development, we will use the definition of end-to-end verifiability given by Popoveniuc *et al.* [11]. They define end-to-end verifiability through the performance requirements set for the voting system. An end-to-end verifiable voting system should provide the following properties:

1. *Cast as intended:* The voter is able to check that her ballot represents a vote for the candidate to whom she intended to give the vote.

2. *Well-formedness:* Anyone is able to check that valid ballots do not contain over-votes or negative votes.
3. *Recorded as cast:* The voter can check that her ballot is recorded as she cast it.
4. *Tallied as recorded:* Anyone is able to check that all the recorded ballots have been tallied correctly.
5. *Consistency:* Anyone is able to check that the voters and the general public have the same view of the election records.
6. *Authenticity/eligibility:* Anyone can check that any cast ballot has a corresponding voter who can perform check No. 3.

Introducing an individual verifiability tool addressed items 1 and 3 above [8]. These are also the two requirements that are targeted towards the voter herself and are hence relatively straightforward to implement.

The remaining four requirements (often referred to as *universal verifiability*) refer to *Anyone* as a potential verifier. The exact meaning of this term is left somewhat vague by Popoveniuc *et al.*, and thus we need to make it clearer before actual implementation.

4.1 From observation to verification

The analogue of universal verification in the case of paper voting is the observation. We design the paper voting methods such as voting in polling stations to protect the integrity of the voting result. With the help of building blocks such as securely sealable ballot boxes, we implement a procedure that makes it possible to claim integrity. As this procedure is carried out by human beings, there is room for mistakes – so as to convince the general public in actual integrity, the observation is applied as a method. Outsiders are allowed to participate and to observe the election procedures – accepting ballots to a ballot box, tabulating the votes, etc. The observers help us to assure the general public that the secure procedure developed for the voting method was correctly put into practice. This assurance makes us trust the integrity of the voting result more.

Following the spirit of universal verifiability, we would like to think that observation is accessible to anybody. This is true in principle, but there are some limitations.

The first obstacle is technical. In case of paper voting, the number of ballots may reach millions and it would be physically inconceivable to recount them all by hand. Instead, the general public relies on a number of designated verifiers (anyone can become one) to check the counting statements to the best of their ability (e.g. partially).

Similarly, verifying statements concerning a digital ballot box or tally integrity assumes proficiency in cryptographic techniques. In principle, anyone can achieve this, but in practice, not everyone does.

The second obstacle involves the threat of coercion. If everyone can get easy access to strong proof that her vote was tallied, this proof can be used to facilitate vote-selling.

The voting legislation in Estonia allows the election organizer to regulate the observation if not all observers can have equal conditions. There is also no universal access to the election data for the observers – for example, lists of voters are out of bounds, only data about the observer herself can be viewed, and the actual tabulation of the votes can only be observed.

The EVIV framework recently proposed by Joaquim *et al.* [9] takes a very pragmatic approach towards universal verifiability. In the election setup, vote-casting and verification phases it explicitly relies on a set of designated trustees, e.g. for distributed key management and homomorphic tally computation.

For the latter task, the EVIV framework introduces a set of trustees (called *Independent Verification Service(s)*). There can be an unlimited number of these services with independent implementations based on the public and verified specifications, the only restriction being the technical/mathematical capability of running the verification. We note that [9] does not specify any conflict resolution mechanisms for cases where some of the services disagree, but at least in principle this task is more feasible compared to agreeing on the count of a pile of paper ballots with possibly millions of ballots.

We shall proceed to present our proposal in the similar trust model as the EVIV. The EVIV uses homomorphic tallying, which has a remarkable performance overhead for large elections (orders of magnitude in Estonia are up to hundreds of candidates in one district and hundreds of thousands of Internet voters). Thus, we will be using provable decryption with mixing to provide vote privacy. However, the problem of verifying the proofs of decryption and mixing still remains, and we will solve it similarly to the EVIV by introducing the Data Auditor role and providing it with cryptographic integrity statements to verify.

This role can be filled by trusted representatives of political parties, foreign research groups or even local civil activists. As is the case with the EVIV, dispute resolution procedures will need to be established in addition to the actual proof-creating software applications.

As with virtually all voting systems with an online component, the Estonian system also features a bulletin board. So far, the functionality of this bulletin board (called *Vote Storage Server*) has been quite restricted, only allowing for limited-time individual verifiability. The second major goal of the current effort is to extend this functionality to also allow the Data Auditor to issue statements about the i-ballot box integrity.

5 IVXV scheme

This Section describes new mechanisms proposed for the Estonian Internet Voting scheme that provide additional mitigation to the threats related to voting result integrity.

Until 2015, external parties were able to observe various organizational procedures during the tabulation process. In the upcoming system (codename IVXV), additional means will be added to verify that the voting result was tabulated

correctly based on the votes that were collected and stored during the voting period.

To perform this verification, a new role – the *Data Auditor* – will be introduced. Technically, the party fulfilling this role will verify the decryption proofs exported during the vote decryption phase. Of course, this process must not violate the secrecy of the votes.

To enable flexible decryption proofs, we replace the current RSA-OAEP cryptosystem used for vote encryption by a randomized homomorphic public-key algorithm, e.g. the ElGamal cryptosystem. This makes it possible to prove correct decryption while preserving privacy, using re-encryption mixnets such as [14], [4] or [2].

To address i-ballot box integrity issues, an extra commitment step will be added. This step will be implemented by a new party called the *Registration Service* that will essentially keep a ledger of the stored i-votes. This makes it, in principle, possible to outsource the duty of collecting votes to a third party, as there is a way to ensure that the integrity of the i-ballot box is maintained.

5.1 Setting

We take advantage of standard cryptographic primitives such as the signature scheme $\sigma = (Gen_{sig}, Sign, Verify)$ with its key-generation, signing and verification functions; the randomized homomorphic public-key cryptosystem $\epsilon = (Gen_{enc}, Enc, Dec)$ with its key-generation, encryption and decryption functions, and the cryptographic hash-function $Hash$.

The Estonian Internet Voting scheme has been using three major components: the Vote Forwarding Server, the Vote Storage Server and the Tabulation Application. In the IVXV, the division is different and signifies an opportunity for the organizational separation of duties. The Voting System is divided between the Election Organizer, the Vote Collector, the I-Ballot Box Processor, the Mixing Service and the Tallier. Additional external parties – the Certification Authority, the Time-marking Service, the Registration Service, the Data Auditor(s) and Voters – interact with the system.

The core requirement for the scheme is the existence of a PKI – there is a Certification Authority CA with the keypair $(sk^{CA}_{pub}, sk^{CA}_{priv})$ and the corresponding certificate $Cert^{CA}_{CA}$.

Eligible voters come from a set of persons where each person has a unique identifier $i \in I$, and everybody is in possession of a signature keypair certified by the CA.

$$\forall i \in I, (sk^i_{pub}, sk^i_{priv}) \leftarrow Gen_{sig}, Cert^i_{CA} = Sign(sk^{CA}_{priv}, (i, sk^i_{pub}))$$

The CA maintains the time-marking service TMS that for any certificate and bitstring pair $(Cert^i_{CA}, b)$ responds with a $Sign(sk^{TMS}_{priv}, (Cert^i_{CA}, b, utc))$ iff the certificate was valid at the time of the request. utc is the time of the request.

There is a Registration Service RS with the keypair $(sk^{RS}_{pub}, sk^{RS}_{priv})$ and the corresponding certificate $Cert^{RS}_{CA}$.

The Election Organizer EO has the duty to determine the voting result. EO approves the election configuration – the PKI and CA, RS, the list of choices C and the list of eligible voters $V \subseteq I$.

The EO selects the encryption system ϵ and generates an election keypair that is used for encrypting and decrypting the votes.

$$(ek^{elec}_{pub}, ek^{elec}_{priv}) \leftarrow Gen_{enc}$$

It is the responsibility of the EO to perform the role of Tallier – to protect the election private key and to tabulate the voting result.

EO provides a Voter with a Voting Application ($VoteApp$) and a Verification Application ($VerApp$). It is assumed that these applications are used on independent devices. The public key ek^{elec}_{pub} is made available to everybody.

EO delegates the handling of the online voting phase to the Vote Collector VC and the handling of the post-voting/pre-tabulation offline phase to the I-Ballot Box Processor $IBBP$. Both VC and $IBBP$ can be independent organizations. EO can nominate a Mixing Service MS.

All voting system components have certified signature keypairs.

We now specify the actions of all roles in the voting process.

5.2 Voting stage

Voting An eligible voter $v \in V$ who wants to vote for a candidate $c_v \in C$ uses $VoteApp$ to create a double envelope.

- The inner envelope is the encrypted choice $ballot_{c,r} = Enc(c_v, r_v, ek^{elec}_{pub})$, where $r_v \leftarrow R$ is a random number.
- The double envelope is acquired by signing the inner envelope digitally with the voter's private key: $vote_v = Sign(sk^v_{priv}, ballot_{c,r})$.
- Voter identifier v, certificate $Cert^v_{CA}$ and double envelope $vote_v$ are sent to the VC.
- VC responds with an unique identifier vid and the RS confirmation reg_{vid}.
- $VoteApp$ verifies the digitally signed reg_{vid} with respect to $Hash(vote_v)$.
- The identifier vid and the randomness used in encryption r_v are presented by the $VoteApp$ in a form that allows them to later be captured by $VerApp$.

Storing the vote In order to store a vote, the VC needs to verify and register the vote.

- VC verifies the eligibility of the voter v and the signature of the vote $vote_v$.
- VC generates a unique random vote identifier vid and stores it together with the vote.

- VC acquires a time-mark $ts_{vid} = Sign(sk_{priv}^{TMS}, (Cert_{CA}^{v}, Hash(ballot_{c,r}), utc_{vid}))$ from the TMS to show that the data $Hash(ballot_{c,r})$ existed at the time utc_{vid} when the voter's certificate was valid. The time-mark is stored together with the vote.
- VC sends a registration request $req_{vid} = Sign(sk_{priv}^{VC}, (vid, Hash(vote_v))$ to the RS.
- RS verifies the registration request, stores it and returns a signed confirmation $reg_{vid} = Sign(sk_{priv}^{RS}, Hash(req_{vid}))$ to the VC.
- VC stores the RS confirmation reg_{vid} together with the vote.
- VC sends the identifier vid and the confirmation reg_{vid} to the $VoteApp$.

If the procedure is a success, the VC stores the following data for a vote: $stored_{vid} = (v, Cert_{CA}^{v}, vote_v, vid, ts_{vid}, reg_{vid})$.

RS stores the $registered_{vid} = (req_{vid}, reg_{vid})$ for each vote.

Note that a voter can cast an i-vote as many times as she likes. All i-votes have to be stored in this phase without removal.

Verifying the vote The voter uses $VerApp$ to check the cast-as-intended and recorded-as-cast properties.

- Voter captures the identifier vid and randomness r_v with $VerApp$.
- $VerApp$ establishes an authenticated TLS channel with VC and sends vid to the VC.
- VC responds to $VerApp$ with a double envelope $vote_v$ and reg_{vid} corresponding to the vid. In case of an unknown vid or exceeded verification timeframe, an error is returned.
- $VerApp$ verifies both the double envelope and RS confirmation. The identity v determined through the verification is displayed to the voter.
- $VerApp$ uses the list of candidates C and the randomness r_v to find a $c' \in C$ such that $Enc(c', r_v, ek_{pub}^{elec}) = ballot_{c,r}$. The result of this process – either the c' or an error message – is displayed to the voter who has to decide if the result represents her will.

The voter is now assured that her vote is both stored and registered correctly.

5.3 Preparing the votes for tabulation

After the online voting phase, the VC contains a set of digitally signed votes D_{VC}, and RS contains a set of registration queries and responses D_{RS}. Both of these sets are transferred to the $IBBP$ responsible for auditing the voting phase and pre-processing the votes for tabulation – revoking superfluous votes, anonymizing votes.

- $IBBP$ verifies all double envelopes, checks eligibility and verifies RS confirmations.

– $IBBP$ compares D_{VC} and D_{RS} for consistency and composes a new list of double envelopes D^1_{IBBP}. This list only contains the latest vote $vote_v$ for each voter v and all entries must have a corresponding registration confirmation linked to the $Hash(vote_v)$.
– $IBBP$ provides EO with the list of people who have i-voted and receives a list of people whose i-vote needs to be revoked, because there is also a corresponding paper vote. $IBBP$ removes those votes from D^1_{IBBP} and gets a new list D^2_{IBBP} as a result.
– $IBBP$ anonymizes the double envelopes in the list D^2_{IBBP}, i.e. extracts the list B_1 of encrypted ballots to be tabulated.

$IBBP$ may pass the list B_1 to EO for tabulation. This is equivalent to the current Estonian Internet Voting. Optionally, $IBBP$ can pass B_1 to the re-encryption mixnet MS in order to cryptographically anonymize the votes. The mixnet shuffles and re-encrypts the input votes B_1 and provides the output set of votes B_2 together with the proof of correct operation P_{mix}.

5.4 Tabulating the voting result

One of the two lists of encrypted ballots – B_1 or B_2 – is passed to the EO for tabulation. The EO uses the election private key to decrypt each choice c' and to calculate the voting result *result*. The EO must also provide a proof of correct decryption P_{dec} together with the plaintext. In case of the ElGamal cryptosystem, a Schnorr identification proof could be used [12].

5.5 Auditing the election

In order to claim the integrity of the voting result, we need to audit the processes that led to that result. We will now show step by step, how an election can be audited in the IVXV scheme.

Auditing *VC* We rely on digital signatures for vote integrity and on individual verification to ensure the cast-as-intended and recorded-as-cast properties. Note that both voting and individual verification steps also check for the correct registration of the vote by RS. Given that the RS and VC are not compromised in a synchronised manner, we can detect the unauthorized removal of votes from the i-ballot box using the following procedure. We define step *AuditVC* for verifying the integrity of the i-ballot box as it is retrieved from the VC.

AuditVC takes D_{VC}, D_{RS} and D^1_{IBBP} as inputs. *AuditVC* accepts iff

– All votes in D_{VC} belong to eligible voters and verify successfully.
– All votes are consistent with the rules of well-formedness.
– All confirmations in D_{RS} verify successfully.
– The views D_{VC} and D_{RS} are consistent.
– The removal of double votes yields D^1_{IBBP}.

Note that due to e.g. network errors there may be votes that are in D_{VC}, but not D_{RS}. The inconsistencies in this step do not mean an immediate problem, but call for further clarification based on e.g. technical logs.

The step *AuditVC* is part of the routine operation by $IBBP$, as the honest operation of the possibly outsourced VC needs to be verified at all times.

Auditing *IBBP* The $IBBP$ makes changes to the contents of the i-ballot box retrieved from the VC – it revokes any votes for a voter v who has voted also on paper, and it only adds to the tally the last i-vote cast by the voter. $IBBP$ also provides a list of encrypted ballots for tabulation.

The $IBBP$ procedure is well-defined and repeatable – the process must always produce the same outputs on the same inputs regardless of implementation. In addition to a complete re-execution of the $IBBP$ procedure, it is possible to perform simple risk-limiting audits – for any vote excluded from the list of votes sent to the tally the $IBBP$ must be capable of providing both VC and RS data together with the reason for revocation.

We refer to the complete auditing step of $IBBP$ as $AuditIBBP$.

Auditing tabulation The optional mixing step performed by the MS and the decryption performed by the EO are verifiable by definition. Given a verifiable re-encryption mixnet and proof of correct decryption, the following sets of data give assurance as to the correctness of the voting result: $(B_1, B_2, P_{mix}, P_{dec}, result)$.

We refer to the auditing step of the MS as $AuditMix$ and the auditing step of tabulation as $AuditTally$.

Complete audit of an election The complete audit of an election that would fulfil the criteria of universal verifiability would consist of all steps: $AuditVC$, $AuditIBBP$, $AuditMix$ and $AuditTally$. Informally, the Data Auditor can be assured of the following properties.

- The integrity of the i-ballot box was preserved.
- The contents of the i-ballot box were processed according to the rules.
- The decryption of a list of encrypted ballots B_2 that is equivalent to the original list of encrypted ballots B_1 was done correctly.

These audit steps achieve the verifiability criteria of [11] as follows.

- *Well-formedness* of the double envelope is verified by the $AuditVC$ and the inner envelope is verified by the $AuditTally$. As we do not apply any proof-technique to show that the encrypted data identified an existing candidate, we may have invalid votes that are only detected during the decryption. We do not consider this to be a problem, as we are not implementing homomorphic tally.
- *Tallied as recorded* is achieved by verifying the i-ballot-box integrity and correct post-processing in the $AuditVC$ and $AuditIBBP$, and verifying the correct tabulation in the $AuditMix$ and $AuditTally$.

– *Consistency* is verified by performing the $AuditVC$ and $AuditIBBP$, and checking that the output of the $IBBP$ process is sent to the MS as input.
– *Authenticity/eligibility* is verified by performing the $AuditVC$ and checking that all the double envelopes were signed by eligible voters.

All these checks have to be performed in a holistic manner – in order to be convinced about e.g. consistency, one has to actually perform the complete audit. This way the Data Auditor can verify the integrity of the voting result without breaking ballot secrecy.

6 Discussion

6.1 Levels of auditing

We described auditing steps that are necessary for the Data Auditor to carry out in order to be convinced about the integrity of the voting result. Different stakeholders could nominate different Data Auditors in order to delegate the verification.

The problem with the complete audit as described above is that the Data Auditor gets access to the complete time-marked set of votes. A malicious Data Auditor could find out whether somebody has re-voted either on paper or online. The information could be abused for coercion. Due to the re-encryption mixnet used, the malicious auditor could not break the ballot secrecy, but we still have to trust the auditor. This implies that we have to define the audit ceremony that mitigates the risk of data abuse by additional means.

A more contained version of the audit would require more trust in the system components. We define a partial audit as consisting of the steps $AuditMix$, $AuditTally$ based on the list of original encrypted ballots B_1 as committed to by $IBBP$. The audit step $AuditVC$ has already been performed by the $IBBP$. This means that the $IBBP$ becomes a trusted party. Due to the well-defined procedure, the actions of $IBBP$ can be double-checked. It is an open question if such a ceremony is feasible that would allow the Data Auditor to trust a partial audit based on the data given by the $IBBP$ – it is basically stating that "there is a set of encrypted ballots that yield the election result, we have to trust the $IBBP$ for authenticity".

It would be possible to implement both partial audits and complete audits in parallel – this would enlarge the set of parties who could commit to the authenticity of the inputs to the partial audit, and the partial audits could be carried out by a much wider audience.

We note that the bar of observation for electronic voting is higher than in the case of paper voting. In case of paper voting, the observer has to be capable of understanding and following the organizational procedures. However, observation of electronic voting requires both computational capabilities and understanding of the cryptographic protocol. Also, the capability to either produce a correct implementation of auditing application or to verify the correctness of an existing one is necessary.

Given these relatively high entry-level requirements, the election organizer cannot rely on the general public providing a reasonable number of protocol participants, but has to give access to verification together with the open specifications and reference implementations. For the sake of completeness a more capable auditor should have the opportunity to implement its own tools based on the aforementioned specifications.

6.2 The role of mixing

The mixnet in the IVXV scheme is only necessary for ensuring ballot secrecy in the case of a third-party Data Auditor. By mixing the encrypted ballots and tabulating the mixed set of encrypted ballots, we assure that two sets are equivalent from the perspective of the voting result, but the one-to-one mapping between two sets is obfuscated. This allows us to give access to the data to an external auditor.

In case of a complete audit, the mixing clearly simplifies the audit ceremony – without mixing, the auditor would have access to both digitally signed encrypted ballots and corresponding plaintexts. Hence, without mixing, the different steps of an audit would have to be separated by other means. In case of a partial audit, the mixing can be considered a safety measure – unless there is a way for the auditor to get the original double envelopes, the plaintexts could not be linked to identities.

There is one party – namely the EO – who by definition has access to the original double envelopes and the election private key. In principle, the EO is capable of breaking ballot secrecy completely. This means that the organizational integrity and private key management are crucial for ballot secrecy – this calls for the threshold scheme – either for the hardware security module activation or threshold decryption.

6.3 Outsourcing the vote collection

The assurance of the voting result integrity and ballot secrecy at the same time under the trust assumptions of the Estonian Internet Voting scheme has required the election organizer to become a technical expert in hosting an online service. The IVXV scheme allows to outsource the vote collection task to a third party, as the correct operation of this party is verifiable by voters, third-party auditors and auditors nominated by the election organizer itself.

The IVXV scheme is designed in a way that the $VoteApp$ should not accept the session unless the VC has responded with the registration confirmation reg_{vid}. Also, the step of individual verification shall verify the correct registration of the vote by RS.

A malicious RS can perform a service denial attack, but in case of other components not co-operating, this attack will be discovered. It is important that the VC stores the RS confirmation – otherwise the RS could drop those confirmations.

A malicious VC could attempt to drop votes after the end of the individual verification time-window, but this would be discovered with the help of the RS that stores the digitally signed requests by the VC. This means that we need to get both the VC and RS datasets for auditing completeness.

6.4 End-to-end verifiability

The IVXV scheme provides mechanisms for both individual and universal verifiability. The individual verifiability tools are available for any voter to use. Access to the data available for central system auditing has to be restricted, though. Only properly anonymized (e.g. cryptographically mixed) data can be given to anyone, whereas the data that links voter identities with other parameters (such as the time of vote-casting or specific encrypted ballots) has to be audited in a controlled environment by designated trustees.

All criteria required by [11] are fulfilled with respect to the aforementioned restriction: well-formedness, consistency, tallied-as-recorded and authenticity/eligibility can be checked by a designated trustee; cast-as-intended and recorded-as-cast can be checked by any voter.

Note that due to the verification of the digital signature in the individual verification tool, the clash-attack [10] is not possible. However, this means that now the $VerApp$ has access to the voter's identity. This assumes that the verification devices are personalized and cannot be shared among untrusting voters. This is a change with respect to the original verification scheme [8]. We argue that this is a reasonable trade-off, since in 2017 personal mobile devices will be much more widespread than they were in 2013.

Hence, we can conclude that the IVXV scheme achieves all the requirements set in [11] to be called end-to-end verifiable.

7 Conclusions and further work

This paper proposed several improvements to achieve the end-to-end verifiability of Estonian Internet voting. In particular, i-ballot box and tabulation integrity have been addressed. Previously, both of these aspects have relied heavily on human control and organizational measures. In the light of the new proposals, it will be possible to offload a lot of this responsibility onto independent external auditors. In principle, it will even be possible to outsource the vote collection part of the central system to a completely untrusted party.

The implementer of the proposed IVXV framework has already been selected and the target is to roll out the system update in time for the local municipal elections due in October 2017.

It is certain that the system development will not end in 2017. The practical try-outs will give us a lot of information about the open issues, e.g. what kinds of conflicts may arise in practice between independent auditor organizations. Resolving these issues will give us a lot of work in future iterations.

A clear separation of roles and their duties opens up the opportunity to apply IVXV also in other elections, not just national elections in Estonia. Implementing this vision also remains a subject for future development.

8 Acknowledgements

This research has been supported by the Estonian Research Council under grant No. IUT27-1.

References

1. Arne Ansper, Ahto Buldas, Aivo Jürgenson, Mart Oruaas, Jaan Priisalu, Kaido Raiend, Anto Veldre, Jan Willemson, and Kaur Virunurm. E-voting concept security: analysis and measures, 2010. http://www.vvk.ee/public/dok/General_Description_E-Voting_2010.pdf.
2. Stephanie Bayer and Jens Groth. Efficient zero-knowledge argument for correctness of a shuffle. In *Advances in Cryptology–EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 263–280. Springer, 2012.
3. Estonian National Electoral Committee. Statistics about Internet Voting in Estonia. http://vvk.ee/voting-methods-in-estonia/engindex/statistics.
4. Prastudy Fauzi and Helger Lipmaa. Efficient culpably sound NIZK shuffle argument without random oracles. In Kazue Sako, editor, *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings*, volume 9610 of *Lecture Notes in Computer Science*, pages 200–216. Springer, 2016.
5. Stephen N Goggin, Michael D Byrne, and Juan E Gilbert. Post-Election Auditing: Effects of Procedure and Ballot Type on Manual Counting Accuracy, Efficiency, and Auditor Satisfaction and Confidence. *Election Law Journal*, 11(1):36–51, 2012.
6. Sven Heiberg, Peeter Laud, and Jan Willemson. The Application of I-Voting for Estonian Parliamentary Elections of 2011. In Aggelos Kiayias and Helger Lipmaa, editors, *VOTE-ID*, volume 7187 of *Lecture Notes in Computer Science*, pages 208–223. Springer, 2011.
7. Sven Heiberg and Jan Willemson. Modeling Threats of a Voting Method. In Dimitrios Zissis and Dimitrios Lekkas, editors, *Design, Development, and Use of Secure Electronic Voting Systems*, pages 128–148. IGI Global, 2014.
8. Sven Heiberg and Jan Willemson. Verifiable Internet Voting in Estonia. In Robert Krimmer and Melanie Volkamer, editors, *6th International Conference on Electronic Voting 2014, (EVOTE 2014), October 28-31, 2014, Bregenz, Austria*, pages 23–29. TUT Press, 2014.
9. Rui Joaquim, Paulo Ferreira, and Carlos Ribeiro. EVIV: An end-to-end verifiable Internet voting system. *Computers & Security*, 32:170–191, 2013.
10. Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Clash attacks on the verifiability of e-voting systems. In *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*, pages 395–409. IEEE Computer Society, 2012.
11. Stefan Popoveniuc, John Kelsey, Andrew Regenscheid, and Poorvi L. Vora. Performance requirements for end-to-end verifiable elections. In Douglas W. Jones, Jean-Jacques Quisquater, and Eric Rescorla, editors, *2010 Electronic Voting Technology*

Workshop / Workshop on Trustworthy Elections, EVT/WOTE '10, Washington, D.C., USA, August 9-10, 2010. USENIX Association, 2010.
12. Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1989.
13. Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the Estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.
14. Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In *Progress in Cryptology–AFRICACRYPT 2010*, volume 6055 of *LNCS*, pages 100–113. Springer, 2010.