VMV: Augmenting Internet Voting Systems with Selene Verifiability using Permissioned Distributed Ledger

Muntadher Sallal[1], Steve Schneider[1], Matthew Casey[3], Catalin Dragan[1], François Dupressoir[1], Jinguang Han[1], Luke Riley[2], Helen Treharne[1], and Joe Wadsworth[4]

[1] University of Surrey, UK
[2] King's College London, UK
[3] Pervasive Intelligence, UK
[4] Formerly Electoral Reform Services Ltd

Abstract. Electronic voting and online voting is an activity that takes place generally independent of verifiability concerns, by organisations that are trusted to provide ballot privacy and correctness of the result. Introducing verifiability through replacing existing systems with brand new algorithms and code presents a risk to businesses. We discuss one approach for incremental change, adding a Selene-based verifiability layer to a TTP-based online voting system, resulting in a fully verifiable and transparent system called Verify My Vote (VMV). Selene is an e-voting protocol that publishes votes in plaintext alongside tracker numbers that enable voters to confirm that their votes have been captured correctly by the system. In addition, this paper describes an experimental setup through which the hybrid system can be evaluated. We also outline how this change supports further incremental changes towards the deployment of fully trustworthy online voting systems. The paper also describes how to use Distributed Ledger Technology as a component of the implementation of Selene to manage the verifiability data in a distributed way for resilience and trust.

Keywords: Verifiable voting · Online voting · Selene · Distributed ledger technology

1 Introduction

Verifiability in e-voting plays an important role in reducing the necessity to trust electronic systems throughout offering both voters and observers an opportunity to independently verify whether votes have been recorded, tallied and counted correctly. However, current internet voting systems are not sufficient to satisfy verifiable elections as they do not provide any proofs or confirming evidences that allow clear individual and universal verifiability [1]. Integrating verifiability with the already existing e-voting systems may require significant changes in the voting experience and structure of these systems which unlikely to be accepted. On the other hand, achieving verifiability within these systems in a way that preserves voters usability is a challenging task to fulfill. Most E2E schemes that support verifiability require voter action such as cut-and-choose during the casting time, making the act of casting the ballot more complex for voters. Therefore, replacing existing systems with a brand new E2E verifiability that supports voters usability presents a risk to businesses.

Aiming to describe how to add the verifaibility as layer on top of existing internet voting systems without significant changes with respect to the voters' experience and overall structure of these systems, Verify My Vote (VMV) approach is introduced in this paper. VMV describes a demonstrator system, currently under development which adds Selene, a brand new end-to-end verifiability protocol, layer to an existing internet voting system used for balloting members by organisations such as trades unions, political parties, professional and building societies. In the VMV, the e-voting scheme Selene [2] is selected to provide the verifiability element. Selene is designed for greater ease of understanding for voters verifying their vote, in that they can see their vote in plaintext when they come to confirm it. Furthermore, Selene proposes an approach which also provides coercion-resistance for the system. It achieves this through commitments to tracker numbers issued to voters, which enable them to look up their vote but such that they cannot prove their tracker number to a coercer. Selene makes use of some sophisticated cryptography to enable this, but this is behind the scenes for the voter. Moreover, Selene does not require any additional voter action during the vote casting time, which simplifies the voter's experience. VMV makes use of distributed ledger technology as a component of the implementation of Selene to manage the commitments in advance of the election, the verifiability evidence produced during the election, and the mixing and decrypting of the votes and tracker numbers at the end of the ballot.

The VMV demonstrator voting system also explores and evaluates the practical issues as well as voters' experience around such systems throughout running trials in the wild, with voters in real ballots. We work with our commercial partner Electoral Reform Services (ERS), a Civica Group company, to introduce verifiability to their existing internet voting system. The system currently has access to the plain votes and is therefore trusted to ensure ballot privacy for voters, so the approached described here still relies on the trusted party for privacy. In order to achieve stronger privacy we would have to remove some element of the information that is held (so that for example votes are only held in encrypted form). At this stage we are adding an additional layer but not removing anything from the existing system.

The paper is structured as follows: Section 2 introduce the standard internet voting system; Section 3 briefly introduces Selene; Section 4 describes the VMV system under development and the considerations that have gone into its design; Section 5 describes VMV from the voter's point of view; Section 6 outlines VMV protocol from technical and cryptographic perspective; finally Section 7 summarises the current state of play and considers which challenges remain outstanding.

2 Internet Voting System

The internet system we are considering relies on a content management system to define the details of an election, and uses this to drive an interactive website to enable voters to be able to log into the website and cast their votes. In practice the voting system is also augmented by other types of non-electronic voting methods such as postal, telephone and sms voting, so that voters have a choice of channels for casting their vote.

The system does not support any form of end-to-end verifiability. It provides the voters with security credentials ahead of the ballot to enable them to log in to a secure system and then cast their vote. Then the cast votes are stored in a secure database effectively in plain text.

2.1 Participants

The key parties in the election process include:

Election Administrator: the organisation responsible for delivering efficient and accurate elections throughout. This is a party who must be trusted by all participants and stakeholders in the election.

Client: requires elections to be held, this might be a union, political parties, building societies, professional organisation, etc. The client role is to provide the information about the election that the client requires, including:
1. the electoral roll, containing the list of all eligible voters;
2. list of choices available for voters;
3. Voting rules to be adopted in the election.

Voter: a person who votes or has the right to vote in an election. A voter is usually a member of the organization for which the election is held.

Trustee: an organization or individual who participates in the administration process of elections with the aim of ensuring that the election runs correctly.

Auditor: an organization or individual that performs auditing, checking the integrity of the election, including accessing the data produced during the election process and performing checks of correctness.

2.2 Supported Types of Elections

Internet voting as provided by ERS supports different kinds of ballots including:

Xvote: In this election the voter is eligible to select one (or possibly multiple) alternatives from a list of candidates.

Resolution Ballot: Each voter responds with a vote of "yes" or "no." for a given question.

Preferential voting: The voter expresses her ranking of the n candidates in order, from 1 (highest) to n (lowest). Partial lists may be allowed. Preferential votes may be tallied in a number of ways, the most commonly used is Single Transferable Vote.

Proxy: The voter delegates her voting power to a representative that can vote on her behalf. In terms of proxy voting, the voter who delegates her vote is known as the principal, while the representative who got delegated is known as proxy. Proxies cannot further delegate a proxy vote.

2.3 Voting Experience

The election is carried out remotely through electronic devices owned by the voters that connect to the election server through a web browser. Voters login into a secure system to populate a secure database with votes. The database stores the cast votes in different string formats based on the type of election. There are a number of phases for the election that involve the voters:

Pre-election phase The initial establishment of the election requires the following to be specified:

Election content: Which identifies the type of elections (e.g. Xvote, Resolution Ballot, Preferential voting, whether Proxy voting is supported).

Choices: The list of candidates or choices that need to be voted on.

Eligible voters: Which indicates voters who are eligible to participate in the election. All voter data is stored within the secure database. On getting a list of eligible voters, Security credentials are generated for each eligible voter with the aim of giving permission only for eligible voters to access the voting system. The security credentials for each voter are sent either by email or postal service.

Election phase During the voting period, the voter uses her credentials to log into the system through a secure web interface. Following successful login, the voter will sequentially pass through several web interfaces towards successfully creating and casting the vote. These web pages include:

1. voting page: the voter is presented with a list of a available choices, that she can select depending on the kind of election and on the voting rule adopted.
2. confirmation page: the voter is asked to confirm their selected choice.
3. Thank you page: the voter gets a confirmation of their vote being correctly recorded.
4. Questionnaire(optional): the voter is asked to answer some user experience related questions.

Post-election phase The result is computed by the Election Administrator and returned to the Client for onward dissemination.

3 The core of Selene

Selene is an e-voting protocol that allows public verifiability by voters/individuals relaying on the idea of assigning a tracking number for each voter. The tracking number is carefully designed where a given tracking number points to a unique and correct vote. At the end of an election, the tracking numbers alongside votes are posted to the BB (Bulletin Board), so each voter can check her vote by simply check the vote against her tracking number. Selene offers an extra property by which *voter coercion threats and vote buying* are mitigated. Coercion mitigation in Selene is fulfilled within two phases. In the first phase, the resistance is achieved based on the idea of the tracking numbers being announced to voters only after the end of elections, so there is no possibility for voters to be coerced during elections as a voter cannot prove her tracking number to a coercer. In the event of coercion (at the end of an election), Selene enables voters to create a fake tracker pointing to the required vote and claim it as theirs.

In order to ensure that each voter is assigned a unique tracker number as well as the notification of the tracking number is done after the vote/tracker pairs have been published, Selene protocol publishes a list of tracking numbers n_i that are encrypted and shuffled. Based on the secret permutation π resulting from the shuffles, the encrypted tracking numbers are assigned for each voter under trapdoor commitments.

$$C_i = h_i^{r_i}.g^{n_{\pi(i)}} \tag{1}$$

Where $h_i^{r_i}$ is the voter's public key which is considered as voter's trapdoor key. $g^{n_{\pi(i)}}$ is the second term β_i of the exponential El-gamal encryption under the voter's trapdoor public key $h_i^{r_i}$. The corresponding α_i term (g^{r_i}) is not published alongside the β_i term. Instead, the α_i term is sent over a private channel to the voter after the election ends. This allows the voter to reconstruct the tracking number only by combining the α_i and β_i term and decrypt the tracking number using the voter's trapdoor key $h_i^{r_i}$.

4 The core of VMV

The VMV protocol aims to achieve individual and universal verifiability without modifying the core voting system, and also without significantly changing the voter's experience. To do so, the Selene e-voting protocol is selected to be as an additional layer on top of the existing system. By using Selene, VMV fulfills two main attractive features. On one hand, the votes cast by voters are collected and made available on a permissioned distributed ledger (DL) in a plain text form and are paired with suitably anonymised tracking numbers. This is in contrast with many other e-voting systems in the literature, where only an encrypted/hashed version of the votes is published on the *BB*. Having plaintext votes published on the DL means voters are not required to trust (or having any particular knowledge of) the cryptographic processes utilized in the election to be confident that their vote was correctly recorded and tallied. Since votes are published in plain text on the DL, individual verifiability is obviously guaranteed (anyone can check their intended vote against the plain text vote recorded in the DL through their tracking number) and so is universal verifiability (anyone can compute the tally from the plaintext votes). On the other hand, VMV also provides a coercion mitigation mechanism, via the Selene protocol, which voters can exploit in case of coercion.

For the verifiability purpose of VMV, a permissioned distributed ledger is used to store encrypted votes alongside the encrypted tracking numbers during the election, and to store plain text votes alongside the anonymised tracking numbers after the election ends. Instead of having only one election authority controlling the whole election and being responsible for publishing the election results, VMV distributes control of an election over several election trustees via a permissioned distributed ledger. This means that the election authorities should all agree on the election information that needs to be published in the distributed ledger, increasing the level of integrity and trust in the election. On the other hand, distributed ledgers offer the additional property where the stored information cannot be changed, so the voter can ensure that her vote recorded during the voting period is never changed or removed during the tally process (after the election finishes). The VMV distributed ledger offers read/write access granted to the election trustees. On the other hand, VMV distributed ledger allows read only access to Voters and Auditors. More details about the permissioned distributed ledger of VMV and what kind of consensus protocol is used will be given in Section 4.2.

4.1 Related work

In the literature, many E2E voting systems have been proposed in relation to in-person voting and internet voting. Examples include Scantegrity II [3], Prêt à Voter [4], Wombat [5] among others. The voting trial of Scantegrity II at Takoma Park was notable in making use of the Bitcoin Blockchain to underpin a commitment to verification information by providing an assurance it was published before the election. To the authors' knowledge this is the first use of Blockchain within a voting system.

Helios [6] provides verifiable voting over the internet for low-coercion environments, and has been used in numerous internet elections for organisations since 2009. Belenios, [7], a more recent scheme, also introduces eligibility verifiability—the ability to verify that all votes have been cast by voters with valid credentials.

In recent years, a number of schemes for applying distributed ledger technology to internet voting have been proposed [8] [9] [10] [11] or launched as systems (e.g. Follow My Vote [5] and Democracy.Earth [6] among many examples). These proposals typically treat the casting of the vote as a transaction to be recorded on a blockchain or distributed ledger, and in some cases casting a vote corresponds to passing a vote "coin" to a particular candidate. However these proposals typically do not address many of the issues around electronic voting such as vote privacy, end-to-end verifiability, coercion-resistance, assurance of cast-as-intended or captured-as-cast, voter eligibility, end-to-end verifiability, and ensuring that all votes cast before the close of the election will be included. Furthermore they might have undesirable side effects such as exposing running totals or turnout in real-time during the election, or associating a monetary value with a vote. In some cases, limited levels of information are available about technical details of their systems is provided so it is difficult to review and independently assess the technical details of the schemes.

In these proposals, the distributed ledger concept has been pushed as a solution for integrity and verifiability problems. However, there are several security and performance issues that occur due to the adoption of permissionless distributed ledger with public peer-to-peer networks, which have not been addressed in these proposals. On one hand, there is no clear definition of necessary and sufficient conditions for independent consistency verification in these proposals which might cause inconsistency in the distributed ledger. In addition, the fact that transactions can be delayed due to the information propagation delay in the public network that maintains the distributed ledger is not taken into consideration. This might result in votes being missing from the distributed ledger at the end of elections. Furthermore, the performance issue that is related to the transaction cost which is quite high in case of using public distributed ledger, is not addressed.

4.2 Distributed ledger

Distributed ledger (DL), also named as blockchain, is simply the ledger of all transactions, grouped into blocks. Every block is linked with previous blocks by including

[5] https://followmyvote.com/

[6] https://www.democracy.ea

the unique hash of the previous block in its header. The first block in the blockchain is known as the genesis block and it has no references to previous blocks. A branch is a path in the blockchain which starts from a leaf block to the genesis block [12]. Blockchain allows an immutable record of all transactions to be created, such that it is resistant to modification from the most resourceful attackers. Valid transactions disseminated in the network that maintains the blockchain, are collected in a block by miners. After that, this block requires a degree of computational effort before it will be accepted by other nodes as valid.

There are several possible types of DLs, which differentiate mainly with respect to two orthogonal characteristics: *(i)* access to the DL and *(ii)* consensus protocol.

As regards access to the DL, there exist two kinds of DL:

Permissionedless DL: public ledger that is not owned by any single authority and, in principle, any one can make "legitimate" (i.e. well-formed according to the rules for the data held by DL) additions to the DL. A permissionedless DL is maintained over a public peer-to-peer network where every node in the network is allowed to contribute data to the ledger and to have identical copies of the ledger [13]. This creates an effective mechanism to obtain *censorship resistance*, i.e. no actor can prevent a transaction from being added to the ledger. Participants maintain the integrity through reaching a consensus about its state. Bitcoin blockchain is an example of the permissionedless DL [14].

Permissioned DL: Permissioned ledgers may have one or many owners, and typically allow a limited number of individuals (individually or collectively/jointly) to add items to the ledger. Such individuals need to acquire permission to do so. When a new record is added, the ledger's integrity is typically checked by a *limited consensus process* which is carried out by a trusted actors, such as government, department, or banks. This makes maintaining a shared record simpler and faster (i.e.requiring less computational resources) than the consensus process used by the Permissionedless DLs. Monax[7], Multichain[8], and Quorum[9] are development platforms that allow to build and manage permissioned ledgers. If more than an individual is allowed to add information to the DL, a consensus protocol is employed to decide which block (among the ones produced by the various nodes in the DL) will be the next to be added to the ledger.

In VMV, we use a permissioned distributed ledger as it seems the best design choice due to the fact that the DL will be purpose-build for the election. So there is no need for individuals other than the ones responsible for conducting the election (which are already trusted in the current setting of VMV) to be allowed write access to it. Therefore, malicious nodes are prevented to take part in the DL's network which maximises the security awareness in the VMV. Furthermore, permissioned DLs allow faster and

[7] https://monax.io/
[8] https://www.multichain.com/
[9] https://www.jpmorgan.com/global/Quorum

lighter consensus than the permissionedless DL. In contrast, consistency in permissionless public ledger is not guaranteed as they support *eventual consensus* where forks are possible and eventually resolved, i.e. such as the branch rule of Bitcoin. Quorum DL platform will be used to create a permissioned distributed ledger in the VMV. Quorum is an enterprise-focused version of Ethereum which offers high speed and high throughput processing of private transactions within a permissioned group of known participants[10]. In VMV, the known participants in the permissioned group will include the election authority and other trusted parties such that they cannot violate the integrity of the Ledger unless a threshold of them collude.

As regards to the consensus protocol, the main approaches considered in the literature are :

Proof of work(POW): The next block to be added is the one produced by the first node to solve a computationally intensive cryptopuzzle linking, via cryptographic hashing.

Proof of stake(POS): The next block is chosen through combination of random section and wealth(the so-called "stake")of the node that creates it.

Proof of Authority(POA): Transactions and blocks are validated by approved accounts, known as validators. Typically, in POA nodes have to earn the right to become validators through reputation(a backlog history of validating "good" blocks). By attaching reputation to identity, validators are incentivises to uphold the transaction process, as they do not wish to have their identities attached to a negative reputation.

In VMV, we use POA consensus protocol as it is better suited to the VMV protocol, since any election requires some trusted authority to manage the election roll, so trust in some authorities is necessary in any case for a voting system. POA is firstly proposed in Ethereum ecosystem for private networks and been used in the Aura and Clique clients [15]. In POA, the responsibility of block creation is distributed fairly over the authorities using a *mining rotation* schema. In this schema, the time is divided into *steps*, each of which an authority is elected as a mining leader.

4.3 Participants and Primitives

The main participants of the VMV protocol are:

The voters V_i: Each voter has a secret signing key associated with the corresponding public verification key vk_i. In addition, the voter has public and secret key pair,pk_i , sk_i for the Selene mechanism, where the latter is stored in a Vote Supporting Device (VSD) e.g. a smartphone or a computer. A voter casts a vote, audit her own ballot, (individually) verifies her own vote.

Election Administrator: *EA* starts the election and is in charge of administrative tasks related to the election, such as setting up the election, issuing credentials for Voters and Trustees, starting the tallying process. *EA* also sends the so-called α_i term to the voter, which can be turned into a tracker for her vote using the secret key sk_i. In VMV, ERS takes over the role of *EA*.

[10] https://www.jpmorgan.com/global/Quorum

Trustee: Individual/organisation that helps ensuring the election is correctly held. A Trustee participates to the shuffling of the ballots and/or to the decryption of the ballots and to the construction of tracker number commitments. Trustees also create the public election key PK_T and threshold share the secret key.

Auditor: Audits the election to ensure its integrity. This includes interacting with the DL to obtain all relevant information related to the NIZKPs generated during the process of the election.

Permissioned Distributed Ledger DL: DL plays the role of the public WBB which is maintained by trustees and election administrator.

In VMV, cryptographic primitives include:

– *Threshold cryptography*. VMV uses El-gamal threshold cryptography without any trusted authority. More specifically, El-gamal encryption key (PK_T) is created in a distributed way over the Trustees, following the key generation protocol proposed in [16], where every trustee T_i will have a share $s_i \in Z_q$ of a secret S. A threshold subset of Trustees are able to decrypt the an El-gamal cipher text with a proof of honest decryption.
– *El-gamal Mix-net*. We use Sako-kilian protocol [17] to shuffle and decrypt pairs of encrypted votes and encrypted tracking numbers. Each pair of an encrypted vote and tracking number are put through a verifiable shuffle performed by the mix-net servers [18]. A threshold set of mix-servers perform a verifiable decryption of these shuffled pairs. Proof of shuffling and correct decryption are uploaded on the DL.
– *Zero Knowledge Proof*. In order to fulfill universal verifiability, the strong form of the Fiat-Shamir transform [19] is used with the aim of obtaining non-interactive proofs of knowledge. We use non-interactive Zero-knowledge proof of correct mixing and shuffling, and correct threshold decryption.
– *Plaintext equivalence tests (PETs)*: PETs performs a public verifiable test where a threshold set of Trustees are able to test whether two cipher text are an encryption for the same plain text without revealing the plain text [20].
– *Pedersen commitment schemes*: A commitment scheme is a cryptographic primitive that transforms a value m into a pair (α, β) where β is the locked box and α is the key such that: *(i)* β reveals no information about m, but *(ii)* together (α, β) reveal m. Also it is infeasible to find an α such that (α, β) reveals $m' \neq m$.

5 Voting Experience

In this section, the VMV protocol is discussed from the voter's perspective. However, the voters need to pass through several phases of VMV protocol in order to be able to create, cast, and verify their vote. These phases of VMV will be described in details in the following subsections.

5.1 Registration

We assume that *EA* (ERS) is in hold of voters' record that indicates voters who are eligible to take part in the election. For setting up the voters ready for voting, each voter needs to register remotely into the VMV system prior to the election. More precisely, eligible voters, as shown in Fig. 1, are able to register in the election by simply interacting with the registration web page which is linked to the parameters initiation system (PI) that is able to import the voter's record as well as generate the verification and election parameters. PI system matches the information supplied by a voter with the eligible voters record in order to identify the voter's record. On indicating the voter's record, the voter eligibility is given and voters' credentials are generated.

Based on Selene, the voter's signing key and voter's trap door public key PK_i are also generated in the registration phase. However, the *EA* needs to know where to contact the voter for the tracker retrieval phase, e.g. an email address, or else the voter will have to have a way of retrieving the second half of the tracker.

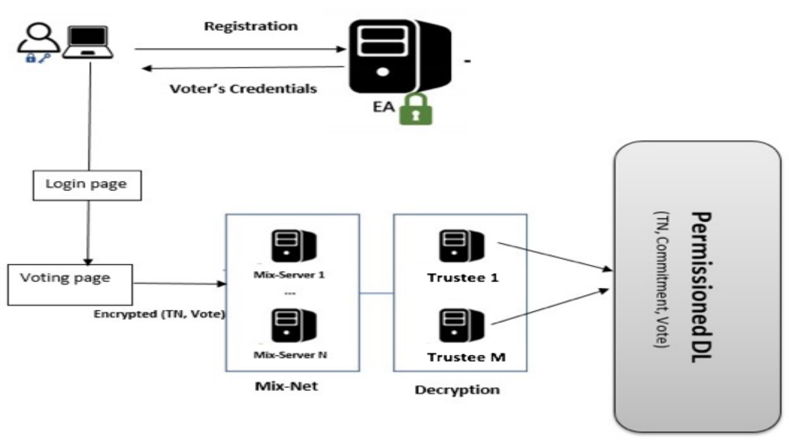

Fig. 1. VMV from voter perspective

5.2 Voting Phase

On the voting day, the *Voting Period* starts where the voter is able to create and cast her vote. More specifically, the voter is asked to input her credentials in order to access the voting web page. In the voting web page, the voter is shown the electoral question and is allowed to select her most preferred alternatives. Once the alternatives have been selected, the voter can go to page Cast Ballot. In this case, an encrypted ballot containing the voter's choices is produced.

In the Cast Ballot page, the voter confirms her choices and the encrypted ballot is sent to distributed ledger. At this point, the voter is issued a tracking number commitment.

5.3 Tracker number retrieval

After the election ends, the cast votes, in plaintext form, alongside the corresponding tracking number will appear on the VMV permissioned DL. If the voter would like to verify that her vote was cast as intended, EA will send the α term, that is generated by Trustees, to the voter via E-mail. On receiving the α term, the voter can open her tracking number commitment, access the information contained in the DL and check her own vote in plaintext (See Fig.2). The voter's unique tracker is calculated by a supported web application using the received α term, the public β term, and the trapdoor key sk. This action is aimed at supporting individual verifiability.

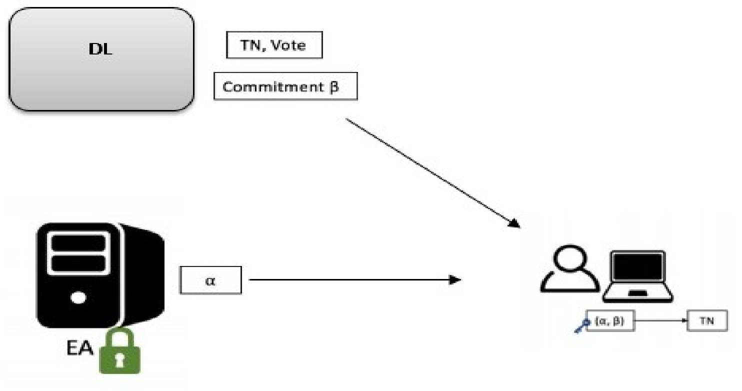

Fig. 2. Description of tracker number retrieval phase

6 Protocol Description

In this section, VMV protocol will be described from technical and cryptographic perspective.

6.1 Pre-Election Setup

In VMV, as in Selene, there are several keys need to be generated prior to any election. These keys include: voter's signing keys , threshold election key, and voter's trapdoor key. On the voter side, signing key and voter's trap door key are generated. We assume that the voter V_i can use the parameter initiator system (PI) to generate El-gamal signing key pair (sk_i, vk_i), where sk_i is the voter's private key and vk_i is the verifier key. Here the voter V_i can produce signatures that can be verified via vk_i. In addition, we assume that PI is able to generate the public key PK_i for the voter V_i, where $PK_i = g^{sk_i}$.

Trustees are responsible for generating threshold election key (PK_T) which is used for El-gamal encryption. The election public key (PK_T) is created in a distributed way over the election Trustees. The encryption public key (PK_T) is generated based on the key generation protocol proposed in Pedersen [16]. In this protocol, every trustee T_i will have a share $s_i \in Z_q$ of a secret S.

As the values $PK_i = g^{s_i}$ are made public, every trustee is committed to these values. So the encryption key PK_T is computed as:

$$PK_T = \Pi_{i=1}^{n} PK_i \tag{2}$$

Where n is the number of trustees. The calculated encryption public key PK_T is sent to all trustees. Note that no single trustee can recover the secret key $S = \Sigma_{i=1}^{n} s_i$.

Turning now to the generation of encrypted tracking number for each registered voter. Each registered voter is assigned an encrypted tracking number as follows:

1. The set of tracking numbers n_i for the registered voters are imported from the election database. These tracking numbers are a sparse selection of integers.
2. g^{n_i} is calculated for each imported tracking number n_i in order to make sure that the tracking number falls in the appropriate subgroup.
3. Using El-gamal encryption scheme, tracking numbers g^{n_i} are encrypted using the encryption key PK_T.
4. Sako-kilian protocol [17] is used to re-encrypt and shuffle $(g^{n_i})_{pk_T}$ in order to offer verifiable re-encryption mixes. Mix-net shuffling is performed on the re-encrypted tracking number. A mix server takes $(g^{n_i})_{pk_T}$ as an input, re-encrypts it using a re-encryption factor s_i and then permutes the encrypted output according to a random permutation π^n. By the end of this action, each voter is associated a unique secret encrypted tracker $(PK_i, \{g^{n_{\pi(i)}}\}_{PK_T})$.

In this stage, trap door commitments that open to a unique trackers are also calculated and attached to the voters' PKs. More specifically, a distributed tracking number commitment for each voter is generated by trustees as follows:

1. Generate a random r_i, where $r_i = \Sigma_{j=1}^{i} r_{i,j}$ for all voters (i) and tellers (j).
2. Generate $\{PK_i{}^{r_i}\}_{PK_T}$, and $\alpha_i = g^{r_i}$
3. Form the product of $\{PK_i{}^{r_i}\}_{PK_T}$ and $\{g^{n_{\pi(i)}}\}_{PK_T}$:
$$\{PK_i{}^{r_i}.g^{n_{\pi(i)}}\}_{PK_T} \tag{3}$$
4. Apply threshold decryption to reveal the commitment:
$$\beta_i = PK_i{}^{r_i}.g^{n_{\pi(i)}} \tag{4}$$
The commitment β_i is produced and published, whereas the $\alpha_i = g^{r_i}$ will be kept secret.

Furthermore, the distributed ledger will be initialized with an initialization block, that will serve as the genesis block, of the chain. The initialization block does not contain any votes, but instead it contains all the information of the election, including the election public key, the set of valid choices the voters can choose from and so on. This way a DL is tied to a specific election and all the system parameters become part of the DL and thus dispute over them is prevented. Security credentials will be generated and given to the Trustees in order to allow them joining the permissioned network and participate in maintaining the distributed ledger (DL). Before the voting phase starts, a tuple of terms are posted to the DL for each voter *i*:

$$(vk_i, PK_i, \{g^{n_{\pi(i)}}\}_{PK_T}, \beta_i) \tag{5}$$

6.2 Voting

To vote, the voter signs in the voting system using their credentials, and is forwarded to the voting page where she can make her choice. After completing the ballot, the voter encrypts the ballot using the election public key, and signs the result using her signature key, and submits the signed encrypted vote.

Voting procedure:

1. Complete the ballot, encrypt and sign it.

$$sign_{v_i}(\{Vote\}_{PK_T}) \tag{6}$$

2. Non-interactive zero-knowledge proof of the knowledge of the plaintext needs (Π_i) to be created and attached with the encrypted ballot.
3. The signed encrypted ballot and the NIZKP are stored in the election database alongside the encrypted tracking number, tracking number commitment, and voter's identity (PK_i), in addition to all the existing voting information recorded by the system.

$$(vk_i, PK_i, \{g^{n_{\pi(i)}}\}_{PK_T}, \beta_i, sign_{v_i}(\{Vote\}_{PK_T}), \Pi_i) \tag{7}$$

4. After the voting period is finished, the signed encrypted ballot and the NIZKP are posted on the ledger alongside the previously published encrypted tracking number, tracking number commitment, and voter's identity (PK_i), as shown in Equation 7. This is the only information published on the ledger; in particular the standard voting information collected by the election system is not published.

6.3 Mix and decryption

For each voter V_i, the encrypted tracker number and encrypted vote are extracted to give a pair of the form:

$$(\{g^{n_{\pi(i)}}\}_{PK_T}, \{vote_i\}_{PK_T}) \tag{8}$$

The set of these is put through a verifiable shuffle performed by the mixnet [18], and then a threshold set of Trustees perform a verifiable decryption of these shuffled pairs. Proof of shuffling and correct decryption are uploaded on the DL. After the decryption process, the following pair is obtained for each voter and published on the DL:

$$(n_{\pi(i)}, vote_i) \tag{9}$$

6.4 Tracking number retrieval

Once the grace period has ended, each voter V_i will be sent the α_i described in Section 3. The received α_i can be combined with the β_i term to reconstruct the tracking number encrypted under the voter's public key PK_i.

$$(\alpha_i, \beta_i) = \{g^{n_{\pi(i)}}\}_{PK_i} \tag{10}$$

So the voter V_i can use her secret key to decrypt and retrieve the $g^{n_{\pi(i)}}$, and thus obtain the tracker $n_{\pi(i)}$, which she can then use to look up her vote on the Ledger.

7 Conclusions and future work

In this paper we proposed VMV, which adds the Selene verifiable voting system onto a deployed internet voting system run by ERS. The main aim of VMV is to provide an end-to-end verifiablity layer on top of an existing voting system. Selene is a fully verifiable and transparent e-voting protocol that publishes votes in plaintext form in the final tally using a tracker system. VMV makes use of a permissioned distributed ledger to manage the verifiability data in a distributed way for resilience and trust. One aspect of ongoing work is to develop a demonstrator voting system to explore the practical issues around such systems.

Integrity: the system design ensures that even an insider at the election authority or an attacker gaining access to the vote database is not able to undetectably change votes, as any change could be detected by the voter attempting a verification. Hence the system no longer requires the authority to be trusted for the election integrity, as it can be verified independently. Use of DLT trustees means that the authority alone cannot change the commitments and verifiability parameters, since collusion between a majority threshold of trustees would be required.

Privacy: the election authority has access to the plain votes and therefore must be trusted for privacy. The election trustees, other than the election authority itself, are not able to see the plain votes, and so they only have access to the information on the Ledger, which protects privacy.

Separation of Duties: The DLT trustees provide assurance of the integrity of the data published on the DLT, and a threshold would be required to tamper with it. The Election trustees provide assurance of the privacy of the vote, and again a threshold of those (or a breach of the vote database) would be required to violate privacy. Both togegther would need to be broken to change the election result in an undetectable way (by changing a vote and also changing the commitment to the tracker number).

There are several ongoing research questions currently being considered. The current demonstrator will only add the signed encrypted votes to the ledger at the end of the election, and they will be collected in the vote database during the election. It may be desirable to publish them in real time to allow voters immediate verification that their information has been posted. However this may give away information during the election such as turnout rate, and the best way of achiving this is currently being investigated.

In practice the system run by ERS also allows for votes to be cancelled through a manual process, removing them from the database of votes. For practical reasons this must be allowed (e.g. if a voter has had her credentials stolen) and we are considering ways to do this in the context of verifiability, non-deletion of data, and how to make use of credentials in issuing replacement ballots.

Our current use of the DLT is as a repository for information, however there is the possibility of using smart contracts to carry out some of the technical elements of verifiability automatically, for example verifying the NIZKPs, or in dispute resolution to carry out some of the necessary checks; this is an area of ongoing work.

Our demonstrator system is currently under development. It will be released as open-source and will be deployed in trials from April 2019.

References

1. Mursi, M. F., Assassa, G. M., Abdelhafez, A., & Samra, K. M. A. (2013). On the development of electronic voting: a survey. International Journal of Computer Applications, 61(16).
2. Ryan, P. Y., Rønne, P. B., & Iovino, V. (2016). Selene: Voting with transparent verifiability and coercion-mitigation. In International Conference on Financial Cryptography and Data Security (pp. 176-192). Springer.
3. Carback, R. et al. (2010). Scantegrity II Municipal Election at Takoma Park: The First E2E Binding Governmental Election with Ballot Privacy, USENIX Security Symposium 2010.
4. Chaum, D., Ryan, P.Y.A., Schneider, S. (2005). A Practical Voter-Verifiable Election Scheme. ESORICS 2005: 118-139
5. Ben-Nun, J. et al. A new implementation of a dual (paper and cryptographic) voting system. In 5th International Conference on Electronic Voting, (EVOTE), 2012.
6. Adida, B. (2008). Helios: Web-based Open-Audit Voting. In USENIX security symposium (Vol. 17, pp. 335-348).
7. Cortier, V., Fuchsbauer, G., & Galindo, D. (2015). BeleniosRF: A Strongly Receipt-Free Electronic Voting Scheme. IACR Cryptology ePrint Archive, 2015, 629.
8. Lee, K., James, J. I., Ejeta, T. G., & Kim, H. J. (2016). Electronic voting service using blockchain. Journal of Digital Forensics, Security and Law, 11(2), 8.
9. Meter, C. (2017). Design of Distributed Voting Systems. arXiv preprint arXiv:1702.02566.
10. Bistarelli, S., Mantilacci, M., Santancini, P., & Santini, F. (2017). An end-to-end voting-system based on bitcoin. In Proceedings of the Symposium on Applied Computing. ACM.
11. Faour, N. (2018). Transparent Voting Platform Based on Permissioned Blockchain. arXiv preprint arXiv:1802.10134.
12. Androulaki, E., Karame, G. O., Roeschlin, M., Scherer, T., & Capkun, S. (2013). Evaluating user privacy in bitcoin. In International Conference on Financial Cryptography and Data Security (pp. 34-51). Springer.
13. Ersoy, O., Ren, Z., Erkin, Z., & Lagendijk, R. L. (2018). Transaction Propagation on Permissionless Blockchains: Incentive and Routing Mechanisms. In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT) (pp. 20-30). IEEE.
14. Fadhil, M., Owen, G., & Adda, M. (2016). Bitcoin network measurements for simulation validation and parameterisation. In 11th International Network Conference, INC 2016. University of Plymouth.
15. De Angelis, S. (2018). Assessing Security and Performances of Consensus algorithms for Permissioned Blockchains. arXiv preprint arXiv:1805.03490.
16. Pedersen, T. P. (1991). A threshold cryptosystem without a trusted party. In Workshop on the Theory and Application of of Cryptographic Techniques (pp. 522-526). Springer.

17. Sako, K., & Kilian, J. (1995). Receipt-free mix-type voting scheme. In International Conference on the Theory and Applications of Cryptographic Techniques (pp. 393-403). Springer.
18. Wikström, D. (2013). User manual for the verificatum mix-net version 1.4. 0. Verificatum AB, Stockholm, Sweden.
19. Fiat, A., & Shamir, A. (1986). How to prove yourself: Practical solutions to identification and signature problems. In Advances in Cryptology—CRYPTO'86 (pp. 186-194). Springer.
20. Jakobsson, M., & Juels, A. (2000). Mix and match: Secure function evaluation via ciphertexts. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 162-177). Springer.