Short paper: Mechanized Proofs of Verifiability and Privacy in a paper-based e-voting Scheme

Marie-Laure Zollinger, Peter B. Rønne, Peter Y.A. Ryan

SnT & University of Luxembourg, Esch-sur-Alzette, Luxembourg
{marie-laure.zollinger, peter.roenne, peter.ryan}@uni.lu

Abstract. Electryo is a paper-based voting protocol that implements the Selene mechanism for individual verifiability. This short paper aims to provide the first formal model of Electryo, with security proofs for vote-privacy and individual verifiability. In general, voting protocols are complex constructs, involving advanced cryptographic primitives and strong security guarantees, posing a serious challenge when wanting to analyse and prove security with formal verification tools. Here we choose to use the Tamarin prover since it is one of the more advanced tools and is able to handle many of the primitives we encounter in the design and analysis of voting protocols.

1 Introduction

In this paper, we propose an initial model for the voting protocol Electryo [12]. Electryo is a paper-based e-voting protocol, where the voter experience remains close to a standard paper-ballot voting scheme, with the Selene mechanism [13] for individual verification.

The additional feature of Electryo is the link between paper ballots and electronic ballots, allowing the possibility to perform (comparison) Risk Limiting Audits [9] efficiently. From Selene it inherits a tracking number feature, allowing voters to verify that their vote has been correctly recorded and counted, but provides a much stronger dispute resolution, as a paper ballot exists which can be compared to the digital record in case of complaint. The voter casts a paper ballot printed at the polling station, which contains an encryption of her ID represented in a QR code. The ballots are scanned to create an encrypted digital version of the paper ballots on the bulletin board. From the data on the bulletin board an anonymous tally list of plaintext votes is created, each associated with a tracking number. After the election ends, each voter will be able to retrieve their tracker and hence check their vote. We will give details of the protocol in section 3.

Among the available tools for formal verification, we chose the Tamarin prover [1] to develop our model. Tamarin has an expressive language based on multiset rewriting rules. This lets us represent a symbolic model of the adversary's knowledge and messages sent over the network. It also uses equational theories, that allow us to specify cryptographic operators, like encryption but also Pedersen commitments. We will detail the semantics of Tamarin and our model in section 4.

Our Contributions. In this paper, we provide a formal model of the Electryo protocol. We model the tracker commitments, encryption, signatures, and channels rules between entities. We provide proofs for ballot-secrecy and individual verification (see section 5). This paper is still a work in progress and additional proofs for receipt-freeness are being developed.

Related Work A TAMARIN model for a simplified version of Selene has already been developed [4]. The authors used the equational theory developed in [7] for the commitments in Selene, which we will use here as well. Vote-Privacy and Receipt-Freeness were proved for a protocol running over untappable channels.

Basin et al. in [3] have developed a protocol for random sample voting with the associated proofs in TAMARIN, and proved Receipt-Freeness and Verifiability. We use their definition of encryption with randomness.

Some other examples of voting protocol models in TAMARIN can be found in [7] where the equational theory for trapdoor commitments have been developed and applied in Okamoto's protocol [11] and the FOO protocol [8].

2 Outline of Selene

We now give a sketch of how voter-verification is achieved in the Selene voting protocol. Full details can be found in [13]. In Selene, the verification is much more direct and intuitive than is the case for conventional End-to-End Verifiability systems: rather than checking for the presence of her encrypted vote on the BB, the voter checks her vote in cleartext in the tally on the BB identified by a secret, deniable tracker.

During the setup phase the set of distinct trackers are posted on the BB, verifiably encrypted and mixed and then assigned to the voters according the resulting secret permutation. This ensures that each voter is assigned a unique, secret tracker. For each encrypted tracker, a trapdoor commitment is created for which the voter holds the secret trapdoor key. In essence this is the "β" term of an El Gamal encryption of the tracker, where the "α" term is kept secret for the moment.

Voting is as usual: an encryption of the vote is created, and sent to the server for posting to the BB against the voter (pseudo)ID. Once we are happy that we have the correct set of validly cast, encrypted votes, we can proceed to tabulation: the encrypted (vote, tracker) pairs are put through verifiable, parallel re-encryption mixes and decrypted, revealing the vote/tracker pairs in plaintext.

Later, the α terms are sent via an untappable channel to the voters to enable them to open the commitment using their secret, trapdoor key. If coerced, the voter can generate a fake α that will open her commitment to an alternative tracker pointing to the coercer's choice. With the trapdoor, creating such a fake α is computationally straightforward. On the other hand, computing a fake α that will open the commitment to a given, valid tracker is intractable without the trapdoor. Thus, assuming that the voter's trapdoor is not compromised, the α term is implicitly authenticated by the fact that it opens to a valid tracker.

3 Electryo

In this paper we only give a very brief overview of Electryo [12] and refer the reader to the original paper for the details. The protocol is summarised in Fig. 1. It is assumed that each voter is equipped with an ID smart card holding their secret signing key and the corresponding verification key is publicly known. Further, Selene requires a separate public key and corresponding secret trapdoor key which the voter holds, e.g. in an app. In the polling station a registration clerk checks the ID card and this is read by a card reader connected to a ballot printer. From the smart card the printer receives an encryption of the ID and an encryption of a signature (e.g. of his PK or ID) from the voter. The printer now delivers a ballot with the re-encrypted ID and signature contained in a QR-Code bcode. Now the voter can fill out the paper ballot in a booth and proceed to a ballot box containing the scanner. The scanner sends the encrypted vote to the bulletin board together with both a re-encryption and a separate encryption of the ballot code. Further, a simple short receipt code is printed and an encryption of this code is sent with the other data to BB. At home the voter needs to enter the receipt code as an authentication token to be able to receive the Selene α term. The receipt code prevents a malicious printer to print another colluding voter's ID on the ballot. The encrypted vote will be associated with the voter via the corresponding encrypted ID, and the Selene mechanism described in the last section can be used.

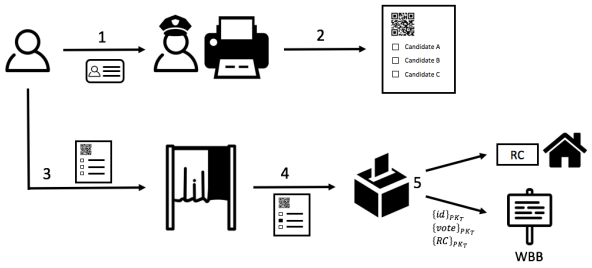

Fig. 1. The Electryo protocol.

4 Tamarin

4.1 Semantics

In Tamarin, messages are represented as terms. A term is an element t or a function $f(t_1, ..., t_n)$ of arity n, where $t_1, \ldots, t_n$ are terms. We also define a set of operators, or functions, with their arities. An equation is a pair of terms s

and t such as $s = t$. We define E as a set of equations. An equational theory is a smallest congruence closure containing all instances E.
Protocols are modeled through multiset rewriting rules. These rules use sets of Facts $F(t_1, ..., t_n)$ of arity n. We denote fresh values with $\sim$ and public values with \$. Facts are user-defined except: Fr, In and Out for inputs and outputs of a rule, and K is the attacker knowledge. An exclamation mark ! before a Fact will define it as persistent and can be consumed many times, while a linear Fact can be consumed only once.

4.2 The Electryo model

In our model, we consider two voters V1 and V2, an Election Authority EA, the Tracker Retrieval Authority TRA and a scanner S. The ID cards, used to perform the ID encryption, are not distinguished from the printer. This is discussed below in the trust assumptions paragraph.

Channel rules We denote by $\bullet\!\!\rightarrow$ an *authentic* channel, that means the adversary cannot modify the messages or their sender, but he can access this data. This ensures that a message is correctly delivered but can be seen and copied by an adversary. More details about TAMARIN channel rules can be found on the manual web page [1].
We also define the *untappable* channel by $\boxed{\longrightarrow}$, which means that a message is not readable nor modifiable over the network like a secure channel [10], but also the message won't be persistent and replayable later.

Trust assumptions In TAMARIN, the adversary is a standard Dolev-Yao style [6], that is controlling the network and can apply all operators. The adversary learns all messages sent by participants when they are output with the Out fact. He can send messages to the participants with the In fact, that is we assume that every input could be given by the adversary. The adversary can also generate fresh values and knows all public values. Finally, he can apply functions available in the set of operators. He will be provided with additional information depending on the trust assumptions below.

As a simplification in the current model, we merged the ID card and the printer into one entity. This means that the printer is reading this voter's ID card without changing the information. The adversary can still modify the information on the printed ballot which will correspond to the ID card and the printer colluding in the original model. We also assume that the voter is using her own ID card which is checked by polling station clerks in the Electryo protocol.

Equational theories To model Electryo, that is using the Selene mechanism, we need the trapdoor (td) commitments equations defined in [7]. This theory is

defined as follows:

$$\begin{aligned}
\mathsf{open}(\mathsf{commit}(m, r, td), r) &= m \\
\mathsf{commit}(m_2, \mathsf{fake}(m_1, r, td, m_2), td) &= \mathsf{commit}(m_1, r, td) \\
\mathsf{open}(\mathsf{commit}(m_1, r, rd), \mathsf{fake}(m_1, r, td, m_2)) &= m_2 \\
\mathsf{fake}(m_1, \mathsf{fake}(m, r, td, m_1), td, m_2) &= \mathsf{fake}(m, r, td, m_2)
\end{aligned}$$

From this equational theory, we can define the trapdoor commitments, which enables voters to fake their tracking number and α term.

We also define an asymmetric encryption scheme. We could not use the existing built-in *asymmetric-encryption* provided by Tamarin because the pre-defined version has no randomness, hence the adversary could learn the encrypted vote sent over the network by using the construction rule for encryption on the public value of candidate. This equation for a ciphertext cp is defined as follows:

$$\mathsf{dcp}(\mathsf{cp}(m, r, \mathsf{pk}(sk)), sk) = m$$

Finally, we use the built-in package *multiset* to model the shuffling of messages as described in [4].

4.3 Tamarin Model of Electryo

An overview of the model is given in Fig. 2. Compared to the existing implementation of Bruni et al. [4], this model considers more cryptographic primitives and provides more data to the adversary as we will detail below. The EA generates tracking numbers and together with the TRA, computes the commitments. The TRA keeps the α-terms secret. The EA publishes the commitments with a persistant fact. Then, voters retrieve their ballot with a ballot code computed from their identity and signature, bcode $= <$ cp($V, r, pkT),cp(sign($V, skV),s, pkT) $>$. Voters input their ballot code and intended vote into the scanner, that computes an encryption of their ballot code *bcode*, an encryption of their vote, generates a receipt-code *RC* and encrypts it. Finally it calculates a re-encryption of the ciphertext buried in the ballot code.[1] In particular, it computes cp($V, r', pkT) and cp(sign($V, skV), s', pkT). The scanner sends all of this data to the EA, and gives the plaintext receipt-code to the voter.
When the EA receives the data for both voters, it decrypts and publishes the votes on the bulletin board with the tracking number and the encrypted RC.
When the votes are published, the TRA can send the α-term to the voter. We use an authentic channel to notify the voters.[2] Each voter can open the commitment and retrieve their tracker. A trace is written to provide a verifiability lemma, checking the validity of the receipt-code and of the vote (see section 5).

[1] For readability, this does not appear in figure 2.
[2] Sufficient for Vote-privacy. To prove Receipt-Freeness, we will need a stronger assumption on channels.

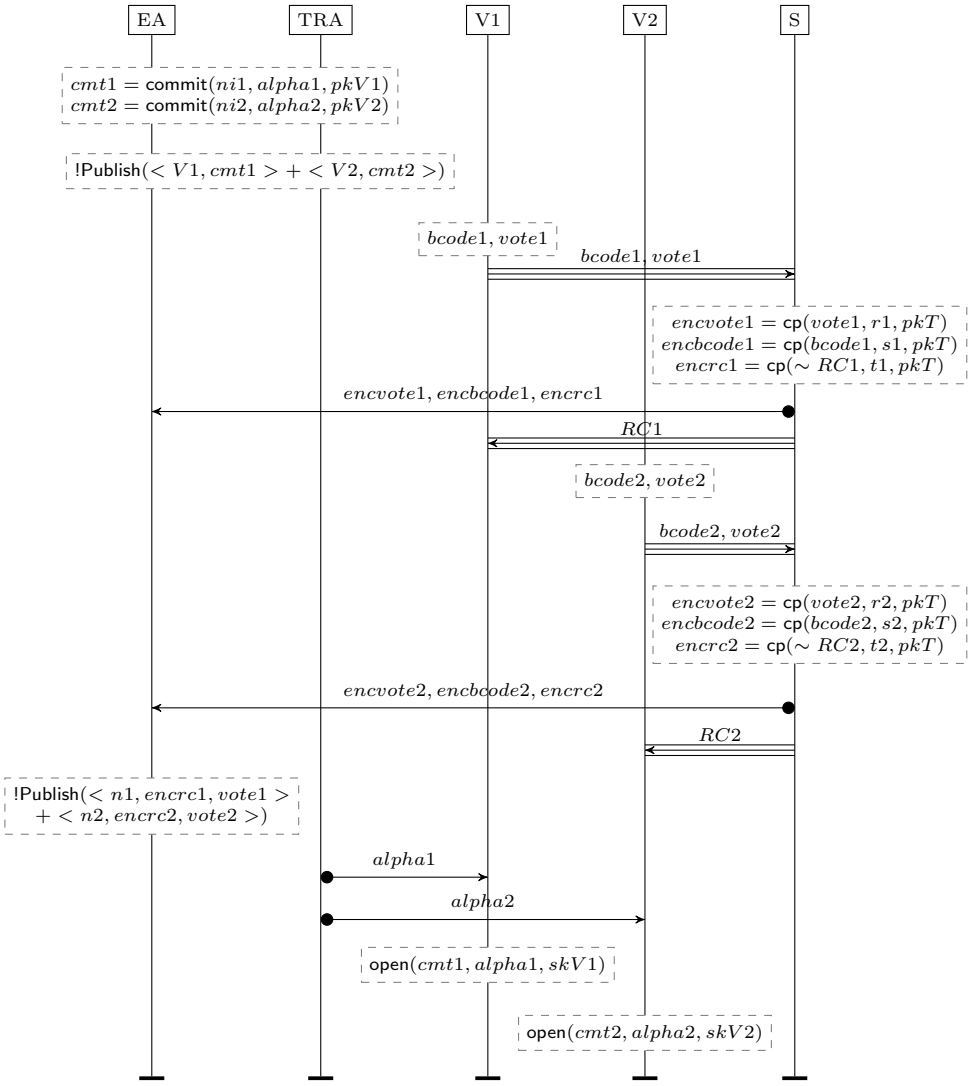

Fig. 2. An overview of the model.

5 Proofs

5.1 Privacy

To prove privacy properties, we need to prove indistinguishability between two executions of a system. In Tamarin, this is done through observational equivalence [2]. For this, the tool uses a multiset rewriting system where terms can be written using a special operator $\mathsf{diff}(\cdot, \cdot)$. With this operator, we are able to

instantiate two possible elements in one term. Then TAMARIN creates two systems, a left and a right, with identical rules where the difference is on the value of the term instantiated with diff.

To verify the observational equivalence, TAMARIN uses dependency graphs. A dependency graph represents the execution of a system. To each node corresponds one rule defined in the model, and there is a direct relation called edge from a rule r_1 to r_2 iff r_1 outputs a fact to r_2 input. The equivalence between two graphs depends on *mirroring*, that is: given a dependency graph, its mirrors contain all graphs on the other side (left or right) of the system defined with the diff operator, where nodes are instances of the same rules and edges are the same.

Vote-privacy First, we used the definition of Delaune et al. [5] for vote-privacy: an attacker cannot detect if voters V1 and V2 swap their votes. In our model, we use the diff operator during the setup phase when defining every entity knowledge: when defining the two voters in the setup rule, we swapped their intended vote. This is defined as follow:

St_V_1('V1', pkV1, ~ltkV1, diff('candA', 'candB'), pkT)
St_V_1('V2', pkV2, ~ltkV2, diff('candB', 'candA'), pkT)

where pkV· is the voter's public key, ~ltkV· is the voter's secret key and pkT is the election key.

In Electryo privacy is guaranteed (for covert adversaries) unless the ID card, printer and the scanner collude. Indeed, we found a trivial attack when the card/printer and the scanner collude. On the other hand, using the above definition, we have a proof in TAMARIN, when neither the card/printer nor the scanner collude with the adversary. Proofs for privacy when only one entity is misbehaving are in progress.

5.2 Electryo verifiability

Verifiability is defined by individual verifiability, that is a voter can verify that her vote was really counted correctly, and universal verifiability, that is the outcome reflects the sum of all cast votes. In this model we only proved individual verifiability. To check verifiability properties, we can use *traces* and express properties as first order logic formulas. These formulas use the temporality of the protocol that let us use the order of events.

In this model, we defined individual verifiability as the ability of voters to correctly check their tracker and verifying that the recorded vote is correct. For Electryo, we also need to verify the correctness of the receipt-code. Given the action Vote, when the voter V casts his vote vote and receives his receipt-code rc, the action Learn when the voter V computes his tracker n, and the action BB, when votes, encrypted ballot-codes and trackers are published, we define individual verifiability as:

```
All V vote rc n #i1 #i2.
    Vote(V, vote, rc) @i1 & Learn(V, n) @i2
==> Ex othervote r pkT #j.
    BB(<n, cp(rc, r, pkT), vote> + othervote) @j
```

To verify this lemma, we used our model defined above, and we modeled a simple malicious behaviour either from the scanner or the card/printer allowing it to modify the ballot-code identity. The lemma remains proven for all traces.

6 Work in progress

A more detailed implementation is still work in progress. In this paper, we provide proofs for verifiability aspects of Electryo with certain channel assumptions. We have shown that an attack from the scanner trying to modify the ballot code is detectable. We have also proven Vote-Privacy of the protocol. We are already using equational theories to model cryptographic primitives (commitments and encryption) but we aim at using these even further in a more detailed modelling of the cryptography. Finally, more proofs regarding privacy properties, in particular Receipt-Freeness, will be the scope of future work.

Acknowledgments We would like to thank the Luxembourg National Research Fund (FNR) for funding, in particular MLZ was supported by the INTER-SeVoTe project and PBR by the FNR CORE project Q-CoDe and INTER-SURCVS. Also the experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg [14] – see `https://hpc.uni.lu`.

References

1. Basin, D.A., Cremers, C., Dreier, J., Meier, S., Sasse, R., Schmidt, B.: Tamarin prover manual. `https://tamarin-prover.github.io/manual/` (2019)
2. Basin, D.A., Dreier, J., Sasse, R.: Automated symbolic proofs of observational equivalence. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (2015)
3. Basin, D.A., Radomirovic, S., Schmid, L.: Alethea: A provably secure random sample voting protocol. In: 31st IEEE Computer Security Foundations Symposium, CSF 2018 (2018)
4. Bruni, A., Drewsen, E., Schürmann, C.: Towards a mechanized proof of selene receipt-freeness and vote-privacy. In: Electronic Voting - Second International Joint Conference, E-Vote-ID 2017 (2017)
5. Delaune, S., Kremer, S., Ryan, M.: Verifying privacy-type properties of electronic voting protocols. Journal of Computer Security (2009)
6. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Trans. Information Theory (1983)

7. Dreier, J., Duménil, C., Kremer, S., Sasse, R.: Beyond subterm-convergent equational theories in automated verification of stateful protocols. In: Principles of Security and Trust - 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS (2017)
8. Fujioka, A., Okamoto, T., Ohta, K.: A practical secret voting scheme for large scale elections. In: Advances in Cryptology - AUSCRYPT '92, Workshop on the Theory and Application of Cryptographic Techniques (1992)
9. Lindeman, M., Stark, P.B.: A gentle introduction to risk-limiting audits. IEEE Security & Privacy (2012)
10. Maurer, U.M., Schmid, P.E.: A calculus for secure channel establishment in open networks. In: Computer Security - ESORICS 94, Third European Symposium on Research in Computer Security (1994)
11. Okamoto, T.: An electronic voting scheme. In: Advanced IT Tools, IFIP World Conference on IT Tools (1996)
12. Rønne, P.B., Ryan, P.Y., Zollinger, M.L.: Electryo, in-person voting with transparent voter verifiability and eligibility verifiability. In: Financial Cryptography and Data Security - FC 2018, Workshop on Voting (2018)
13. Ryan, P.Y.A., Rønne, P.B., Iovino, V.: Selene: Voting with transparent verifiability and coercion-mitigation. In: Financial Cryptography and Data Security - FC 2016 (2016)
14. Varrette, S., Bouvry, P., Cartiaux, H., Georgatos, F.: Management of an academic hpc cluster: The ul experience. In: Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014) (2014)