Proving e-voting mixnets in the CCSA model: zero-knowledge proofs and rewinding

Margot Catinaud Caroline Fontaine Guillaume Scerri

With the ever more widespread use of e-voting for professional and even political elections, developping robust proofs of these protocols is essential. A crucial step in these protocols is mixing the "ballot box" in order to hide the identity of the voters in the tally. While for relatively simple tally functions this can be achieved through homomorphic encryption (computing the encrypted tally and only decrypting the tally), for complex tally functions the use of so called *mixnet* is required.

Mixnet takes as input a list of encrypted votes (potentially together with proofs of correctness of said votes), and produces as output a permutation of (a reencryption of) this list of votes. We expect two security properties from these constructions:

- *Permutation secrecy*: An honest mixnet should ensure that no adversary can link votes in the output list with votes in the input list.
- *Verifiability*: A dishonest mixnet should not be able to convince an honest verifier that the output list is a reencrypted permutation of the input list when it is not the case.

Mixnets typically rely on advanced cryptographic constructions, such as zero knowledge proofs and commitment schemes (with some non-trivial algebraic properties), and complex interactions between these constructions. As a result the proof techniques required for these protocols are quite involved. They typically both need non-standard cryptographic reductions (e.g. rewinding), and complex interactions between algebraic and cryptographic properties.

Considering the complexity of these proofs, it is hard to fully trust handmade cryptographic proofs. As an example the original proof of the Terelius-Wikström [13] mixnet (a variant of which is used in Belenios [8] and CHVote [10]) is far from a full cryptographic proof, but only provides key cryptographic and algebraic ideas. Providing a full, precise cryptographic reduction from this work would be non-trivial. Considering the size and complexity of the resulting proof, its formal verification would yield much higher confidence.

A number of tools aim at mechanizing proofs of cryptographic protocols. Purely symbolic tools (ProVerif [5] and Tamarin [11] for example) do not allow the fine grained probabilistic reasoning needed to faithfully capture the essence of proofs of mixnets. This leaves us with tools that aim at mechanizing proofs in the computational model (i.e. where the adversary is a probabilistic polynomial time Turing machine). The automated tool CryptoVerif [6] does not support arbitrary mathematical reasoning, nor does it support rewinding, which are both necessary for performing proofs of mixnets. Tools based on probabilistic Hoare logic (mainly EasyCrypt [4]), would be more suited for such proofs. While recent advances allow rewinding in EasyCrypt [9], performing complex proofs of protocols using advanced cryptographic techniques is still complex and time consuming in these tools, making them ill suited for our goal.

In this work we focus on a third avenue, namely the Computationally Complete Symbolic Attacker (CCSA) model [3], [2]. This model is based on a logic with a probabilistic semantics. The central predicate of the logic $u \sim v$ encodes the fact that the probability of a probabilistic polynomial time adversary distinguishing the computational interpretation of the terms u and v is negligible. In order to perform a proof in this model, one has to provide elementary axioms in the logic representing the properties of the cryptographic construction used (whose computational interpretation should be proven sound) and then perform the proof based on those axioms. A proof then provides guarantees against a computational attacker. This logic has been implemented in the Squirrel proof assistant [1]. This logic allows relatively simple proofs of complex protocols (for example key-management APIs [12]), with very limited work on proving soundness of the axioms. Notably the soundness proofs are small and relatively easy to check, and the remainder of the reasoning can be checked automatically.

Our main contribution is a proof of the Terelius-Wikström mixnet in the CCSA logic. This required new axioms and an extension of the semantics of the logic. More precisely:

- We provide (and prove) axioms for the algebraic properties needed for the proof.
- We provide the first axiomatization of zero-knowledge proofs, commitment protocols and reencryption.
- We provide a new construction that allows us to capture rewinding in the CCSA logic. This last point is the most complex one. Natively, the original CCSA logic only allows reasoning on globally negligible (or globally non negligible) events, in the sense that one can only reason on probabilities on the whole sampling space, but does not have any construction for dealing with conditional probabilities. However, rewinding requires reasoning on the probability of a certain event *knowing* that an execution point has been reached. We introduce a new construction in the CCSA logic that captures that a certain formula is true with non negligible probability knowing that another formula is true, and provide axioms on the interaction between this construction and the usual global CCSA formulae that allows us to capture the rewinding argument.

Finally we structure the proof in relatively independent lemmas, with clear hypotheses. This gives us confidence that the proof we obtain is rather modular, in the sense that switching one kind of e.g. zero knowledge proof for another (e.g. NIZK instead of sigma protocols) should only minimally affect the proof. Indeed, thanks to the CCSA approach, if the hypothesis of a lemma is the consequence of two different set of axioms (e.g. one for NIZK and one for sigma protocols), the remainder of the proof holds regardless of which cryptographic primitive was used. Actually taking advantage of this modularity, together with implementing our new axioms and constructions in the Squirrel proof assistant is left as future work.

REFERENCES

[1] David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutsos, and Solène Moreau. An interactive prover for protocol verification in the computational model. In *IEEE Symposium on Security and Privacy*, pages 537–554. IEEE, 2021.

[2] David Baelde, Adrien Koutsos, and Joseph Lallemand. A higher-order indistinguishability logic for cryptographic reasoning. In *LICS*. ACM, 2023.

[3] Gergei Bana and Hubert Comon-Lundh. A computationally complete symbolic attacker for equivalence properties. In *CCS*, pages 609–620. ACM, 2014.

[4] Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. Computer-aided security proofs for the working cryptographer. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 71–90. Springer, 2011.

[5] Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14 2001), 11-13 June 2001, Cape Breton, Nova Scotia, Canada*, pages 82–96. IEEE Computer Society, 2001.

[6] Bruno Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Trans. Dependable Secur. Comput.*, 5(4):193–207, 2008.

[7] Véronique Cortier, Constantin Catalin Dragan, François Dupressoir, and Bogdan Warinschi. Machine-checked proofs for electronic voting: Privacy and verifiability for belenios. In *CSF*, pages 298–312. IEEE Computer Society, 2018.

[8] Véronique Cortier, Pierrick Gaudry, and Stéphane Glondu. Belenios: A simple private and verifiable electronic voting system. In *Foundations of Security, Protocols, and Equational Reasoning*, volume 11565 of *Lecture Notes in Computer Science*, pages 214–238. Springer, 2019.

[9] Denis Firsov and Dominique Unruh. Reflection, rewinding, and coin-toss in easycrypt. *IACR Cryptol. ePrint Arch.*, page 1078, 2021.

[10] Rolf Haenni, Reto E. Koenig, Philipp Locher, and Eric Dubuis. Chvote system specification. *IACR Cryptol. ePrint Arch.*, page 325, 2017.

[11] Simon Meier, Benedikt Schmidt, Cas Cremers, and David A. Basin. The TAMARIN prover for the symbolic analysis of security protocols. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *Lecture Notes in Computer Science*, pages 696–701. Springer, 2013.

[12] Guillaume Scerri and Ryan Stanley-Oakes. Analysis of key wrapping apis: Generic policies, computational security. In *CSF*, pages 281–295. IEEE Computer Society, 2016.

[13] Douglas Wikström. A commitment-consistent proof of a shuffle. In *ACISP*, volume 5594 of *Lecture Notes in Computer Science*, pages 407–421. Springer, 2009.