Modularity of mixserver security proofs in the CCSA logic: case of Bayer-Groth protocol

Margot Catinaud Caroline Fontaine Guillaume Scerri

Mixnets protocols are a crucial component of electronic voting protocols as an alternative to homomorphic tally. When the voting system is more complicated than a simple yes/no vote or a first-past-the-post system, homomorphic operations are inadequate. Beyond their use in e-voting protocols, mixnets also have an interest on their own, as soon as we have to shuffle lists of ciphertexts while ensuring cryptographic security properties. Informally, a *mixnet* is a network of mix-servers taking as input a list of ciphertexts. Then, each mix-server shuffles its input list turn accordingly to some specified order. Typically, each mix-server outputs a permuted and re-encrypted version of its input list. Depending on the use-case, mixnets have to ensure *integrity*, *privacy*, *verifiability* and/or *accountability*.

- *Integrity*: For this property, we add an abstract predicate $\mathbf{valid}_n$ over a list of ciphertexts to express some *validity property* of these ciphertexts. Using this predicate, either in case of honest and dishonest mixnet, if the input list is valid in the sense of the $\mathbf{valid}_n$ predicate, then the output list of ciphertexts remains valid;
- *Privacy*: In the case of an honest mixnet, no adversary should be able to link output and input elements;
- *Verifiability*: No adversarial mixnet must be able to fool an honest verifier on its behavior: an honest verifier should detect whether the adversarial mixnet misbehaves or not;
- *Accountability*: This property is a refinement of the verifiability: an honest verifier should be able to identify and blame a mix-server misbehavior.

To achieve such security goals, mixnets rely on advanced cryptographic constructions, such as *zero-knowledge* proofs and commitment schemes, with some non-trivial algebraic interactions between them. In terms of computational security proofs, these constructions solicit involved proof techniques such as *rewinding*, and ask to juggle between cryptographic and probabilistic properties.

As a side note, even if cryptographic properties are probabilistic, the nature of cryptographic and probabilistic properties are completely different. On one hand, cryptographic properties do not require anything about probabilistic distributions of random computations: all computations must be produced by *probabilistic and polynomial-time Turing Machines* (that we abbreviate as PPTM in the rest of the paper). On the other hand, probabilistic properties encompass nothing about how computations are produced: all concerned random variables must be independently and uniformly chosen.

Concerning mixnets used in e-voting protocols, two main shuffle protocols are used. For the case of Belenios [8] or CHVote [9] protocols, the shuffle protocol used is the one of Terelius & Wikström [12]. Another shuffle protocol, designed by Bayer & Groth [4], is used in the SwissPost [11] protocol.

The diversity of kinds of properties and fine-grained probabilistic reasoning needed to prove mixnet security make such proofs complex in terms of reasoning and quite huge in terms of size. Therefore, a formal verification would yield to much higher confidence than pen-and-paper proofs. In the formal verification of cryptographic protocols, two main model are used: the symbolic one and the computational one, both coming with specific tools. Because of involved kinds of properties, purely symbolic tools (such as ProVerif [5] or Tamarin [10]) will not be suitable. This leaves us with computational oriented tools such as CryptoVerif [6], Easycrypt [3] or Squirrel [1].

Recent advances [7] in the *Computationally Complete Symbolic Attacker* (CCSA) model [2] used by the Squirrel prover have been made to catch rewinding argument and used it to prove security of a mixnet based on the Terelius-Wikström shuffle protocol. However, even if axiomatization of rewinding was a complex task, its use in the security proofs of this latter shuffle protocol was quite straightforward because of its straight specification. Indeed, the proof of shuffle in the Terelius-Wikström case firstly proved that the mix-server has committed to a permutation and secondly proved consistency of the shuffle with respect to this commitment. Each of these two proofs were calling rewinding separately. On the contrary, in the present paper, we prove the security of the Bayer-Groth shuffle protocol by using rewinding in a more subtle and intricated way challenging the formal CCSA framework proposed in the previous work mentionned above. More precisely (see fig. 1), the final proof of shuffle is split into two subproofs called after three messages (a first commit message α_1, a challenge x and then a second commit message $\alpha_2(x)$). The first subproof links both commit messages α_1 and $\alpha_2(x)$ while the second subproof only depends on the second commit message $\alpha_2(x)$. Besides, to prove verifiability of the Bayer-Groth protocol, one has to rewind on the challenge x, and thus both subproofs cannot be treated separately.

Another contribution of this paper is to set a formal framework for mix-servers which encompasses both the Terelius-Wikström and Bayer-Groth shuffle protocols. Next, based on this formal framework and on mix-server's security properties, the goal is to prove security properties – mentionned at the beginning – of the global mixnet.

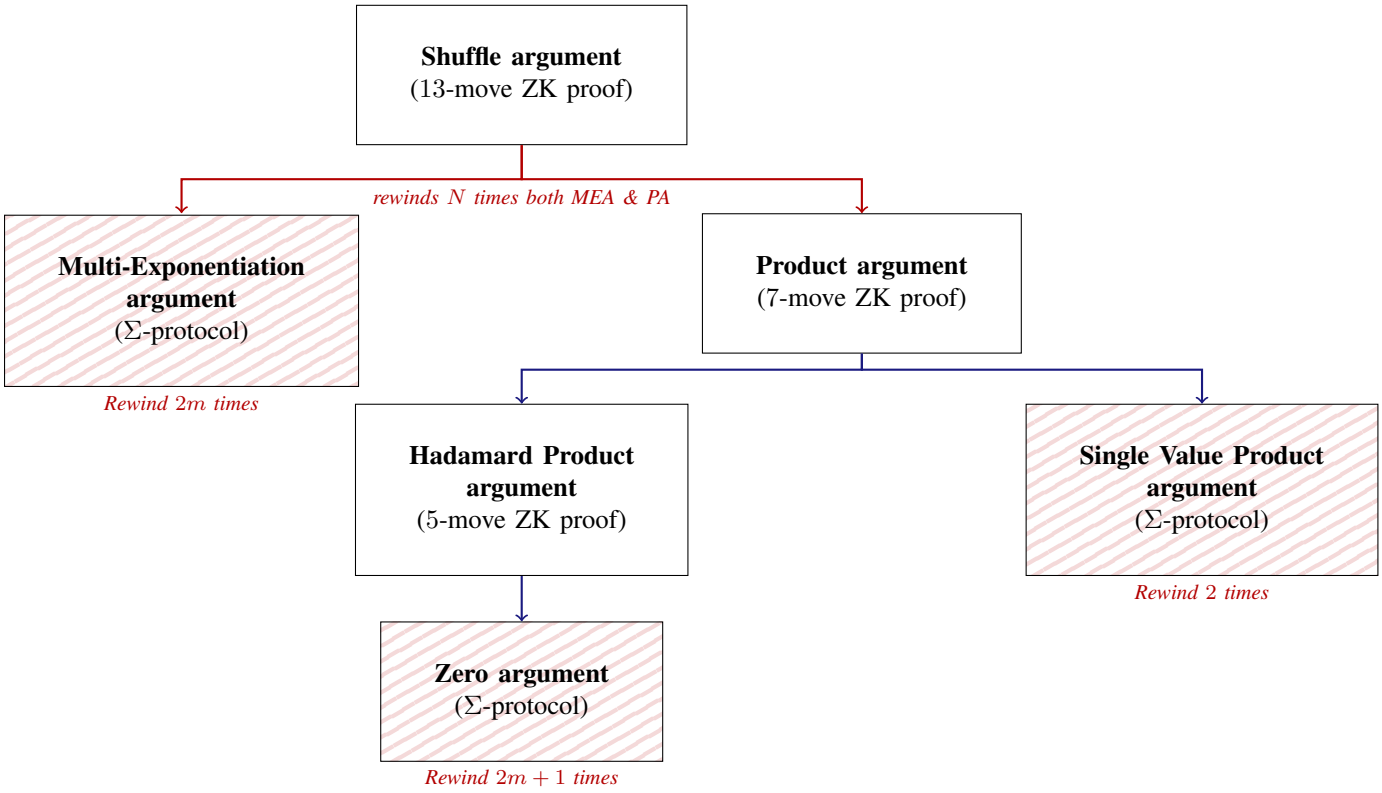

Fig. 1. Structure of the Bayer-Groth shuffle protocol (with $N = nm \in \mathbb{N}$, the length of the input list of ciphertexts)

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] 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.

[4] Stephanie Bayer and Jens Groth. Efficient zero-knowledge argument for correctness of a shuffle. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 263–280. Springer, 2012.

[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] Margot Catinaud, Caroline Fontaine, and Guillaume Scerri. Proving e-voting mixnets in the ccsa model: zero-knowledge proofs and rewinding. 2026.

[8] Stéphane Glondu. *Belenios specification*. Version 3.0.

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

[10] 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.

[11] Swiss Post. *Swiss Post Voting System – System specification*, 2025. Version 1.5.2.

[12] Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In *AFRICACRYPT*, volume 6055 of *Lecture Notes in Computer Science*, pages 100–113. Springer, 2010.