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

Margot Catinaud * Caroline Fontaine * Guillaume Scerri *

*Université Paris-Saclay, CNRS, ENS Paris-Saclay,
Laboratoire Méthodes Formelles (LMF)

GT MFS, April 2024

Laboratoire
Méthodes
Formelles

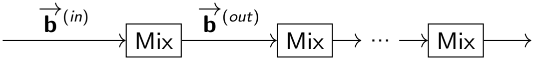
Electronic voting mixnets

Two kinds of tally

Homomorphic encryption

Mix networks + Decrypt

Principle



Network of *mix-servers*

Algorithm : Mixing

let `mixing` $\overrightarrow{\mathbf{b}}^{(in)}$ **=**
  $\pi \xleftarrow{\$} \mathfrak{S}_N$;
  [*do some stuff...*] ;
  return $\overrightarrow{\mathbf{b}}^{(out)}$

Mix-server in a nutshell

Terelius & Wikström mixnet ([TW10], [Wik11])

Security properties for one mix-server

Permutation secrecy

Verifiability

Key ingredients needed

Commitment scheme

Zero-knowledge proofs

Zero-knowledge proofs - case of Σ-protocols

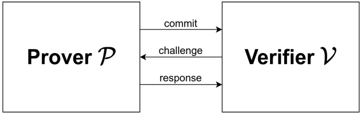
Principle

Two agents: a prover $\mathcal{P}$ and a verifier $\mathcal{V}$

Goal: prove that $(\underbrace{x}_{\textit{statement}}, \underbrace{w}_{\textit{witness}}) \in \mathcal{R}$

Interactive proof: proof transcript

$$(\underbrace{p_0}_{\textit{commit}}, \underbrace{c}_{\textit{challenge}}, \underbrace{p_1}_{\textit{response}})$$



Sigma-protocol

Main security properties

Special-Soundness

Zero-knowledge

Verifiability game

Cryptographic game — Mix-server verifiability.

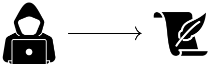
Context

Adversarial mix-server

Honest verifier $\mathcal{V}$

Game statement

Hypothesis



Proofs accepted by $\mathcal{V}$

$\Longrightarrow$

Conclusion

$$\text{Dec } \overrightarrow{\mathbf{b}}^{(out)} = \text{Dec}\left(M_\pi \cdot \overrightarrow{\mathbf{b}}^{(in)}\right)$$

Output plaintexts is a permutation of input

Computationally Complete Symbolic Attacker (CCSA) model

The SQUIRREL prover ([Bae+21])

First introduce by Bana & Comon ([BC14]), high-order logic by Baelde, Koutsos & Lallemand ([BKL23])

Main predicates: $\sim$ (indistinguishability)
and $[\cdot]$ (globally (non-)negligible events)

Interpretation of terms for a *fixed* random tape ρ: $[\![t]\!]_\rho$.

In our case: work on trace properties

Formulas ϕ are terms of type **bool**.

Two kinds of logic

Global logic

$[\phi] \overset{\sim}{\rightarrow} [\psi]$ means:
If $\mathrm{Pr}_{\rho\in\Omega}\left([\![\phi]\!]_\rho\right)$ is overwhelming
then $\mathrm{Pr}_{\rho\in\Omega}\left([\![\psi]\!]_\rho\right)$ is overwhelming.

Local logic

$[\phi \rightarrow \psi]$ means:
$\mathrm{Pr}_{\rho\in\Omega}\left([\![\phi \rightarrow \psi]\!]_\rho\right)$ is overwhelming.

Sketch of proof

Extraction of sealed matrix M

Witness extractor — Rewinding

Collect enough witness — Rewinding

Reconstruction of sealed informations — Algebra

Is M a permutation matrix?

Witness extractor — Rewinding

Witness consistency — Cryptography

Generalization of equations on witness to equations on matrix — Algebra

Characterisation of permutation matrix — Algebra

$\overrightarrow{\mathbf{b}}^{(out)} = \mathrm{ReRand}\left(M \cdot \overrightarrow{\mathbf{b}}^{(in)}\right)$?

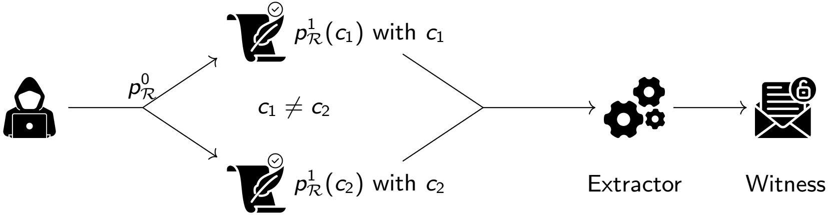
Another witness extractor — Rewinding

Consistency between the witness and the extracted matrix — Cryptography

Generalization to the whole set of ciphertexts in/out pairs — Algebra

Special-Soundness

Statement



Axiomatization in the CCSA logic

L.Σ-P:SpSound

$$\tilde{\exists}\mathbf{zkp\text{-}extract}_{\mathcal{R}}\ [\text{ptime}].[\bigwedge_{i\in\{1,2\}} \mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \underbrace{(p^0_{\mathcal{R}}, c_i, p^{1,(i)}_{\mathcal{R}})}_{\mathfrak{p}^{(i)}_{\mathcal{R}}}) \wedge c_1 \neq c_2 \rightarrow (x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}^{(1)}_{\mathcal{R}}, \mathfrak{p}^{(2)}_{\mathcal{R}})) \in \mathcal{R}]$$

Witness extraction algorithm

Algorithm : Witness extraction

Input: Adversary $\mathcal{A}$ producing sometimes a proof accepted by the verifier $\mathcal{V}$.

Run $p_0 \leftarrow \mathcal{A}(x)$;

repeat

  Choose $c_1 \leftarrow \mathcal{V}(x, p_0)$ then run $p_1 \leftarrow \mathcal{A}(x, p_0, c_1)$;

  Rewind $\mathcal{A}$;

  Choose $c_2 \leftarrow \mathcal{V}(x, p_0)$ then run $p_2 \leftarrow \mathcal{A}(x, p_0, c_2)$;

  Check if $\top \leftarrow \mathcal{V}(x, \mathfrak{p}_1)$ and $\top \leftarrow \mathcal{V}(x, \mathfrak{p}_2)$;

until $\mathfrak{p}_1$ *and* $\mathfrak{p}_2$ *are accepted by* $\mathcal{V}$ *and* $c_1 \neq c_2$;

return $w \leftarrow \textit{zkp-extract}_{\mathcal{R}}(x, \mathfrak{p}_1, \mathfrak{p}_2)$;

where $\mathfrak{p}_i \stackrel{\text{def}}{=} (p_0, c_i, p_i)$ for $i = 1, 2$.

First attempt

A first local hunch...

$$\frac{\textbf{zkp-verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1))}{(x, \textbf{zkp-extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1), \mathfrak{p}_{\mathcal{R}}(r_2))) \in \mathcal{R}}\ \textsc{L.Extract}$$

where $\mathfrak{p}_{\mathcal{R}} \overset{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

First attempt

A first local hunch...

$$\dfrac{\text{L.EXTRACT}\qquad \mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1))}{(x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1), \mathfrak{p}_{\mathcal{R}}(r_2))) \in \mathcal{R}}$$

where $\mathfrak{p}_{\mathcal{R}} \stackrel{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

Problem

$\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1)) \not\Longrightarrow \mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_2))$ for $r_1 \neq r_2$:

First attempt

A first local hunch...

L.Extract

$$\frac{\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1))}{(x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r_1)), \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r_1)))) \in \mathcal{R}}$$

where $\mathfrak{p}_{\mathcal{R}} \stackrel{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

Problem

$\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1)) \not\Longrightarrow \mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_2))$ for $r_1 \neq r_2$:

First attempt

A first local hunch...

L.EXTRACT

$$\frac{\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1))}{(x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r_1)), \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r_1)))) \in \mathcal{R}}$$

where $\mathfrak{p}_{\mathcal{R}} \stackrel{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

Problem

$\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_1)) \not\Rightarrow \mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r_2))$ for $r_1 \neq r_2$:

If ϕ is locally true, it says nothing about the distribution of $\left[[\![\phi]\!]_\rho \mid \rho \in \Omega \right]$.

Thus, we need to characterize events which holds with non-negligible probability.

An addition to the CCSA logic: $_e[\_]$ predicate

$_e[\_]$ predicate

For a formula ϕ : **bool** and a non-negligible term e : **real** [non-negl], we define:

$$_e[\phi] \iff \Pr_{\rho \in \Omega}\left(\llbracket \phi \rrbracket_\rho\right) \geqslant e$$

We want the following equivalence:

$$\tilde{\neg}[\neg\, \phi] \tilde{\Leftrightarrow} \tilde{\exists} e : \textbf{real}\ [\text{non-negl}].\ _e[\phi]$$

and we want

$$_e[\phi(r)] \tilde{\rightarrow} [\phi(\mathbf{resample}(r))]$$

e : **real** [non-negl] means that $\eta \longmapsto \llbracket e \rrbracket^\eta$ is non-negligible,

i.e. their exists a polynomial P such that: $\exists\, \eta_0 \in \mathbb{N}^*, \forall\, \eta > \eta_0, \llbracket e \rrbracket^\eta \geqslant \dfrac{1}{P(\eta)}$.

Are we done yet?

G.Extract

$$\frac{{}_{e}[\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r))]}{[(x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r)), \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r)))) \in \mathcal{R}]}$$

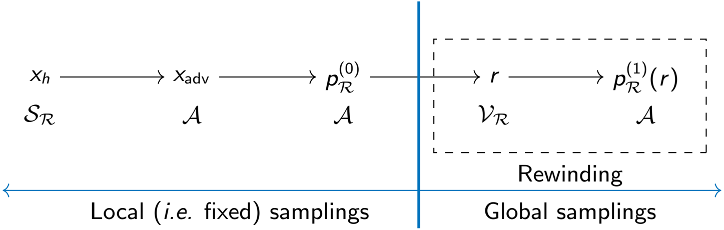
where $\mathfrak{p}_{\mathcal{R}} \stackrel{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

Are we done yet?

$$\textsc{G.Extract}\ \frac{{}_e[\mathbf{zkp\text{-}verif}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(r))]}{[(x, \mathbf{zkp\text{-}extract}_{\mathcal{R}}(x, \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r)), \mathfrak{p}_{\mathcal{R}}(\mathbf{resample}(r)))) \in \mathcal{R}]}$$

where $\mathfrak{p}_{\mathcal{R}} \stackrel{\text{def}}{=} \lambda r.(p_{\mathcal{R}}^{(0)}, r, p_{\mathcal{R}}^{(1)}(r))$ for some *fixed* $p_{\mathcal{R}}^{(0)}$.

No, not yet



What is missing

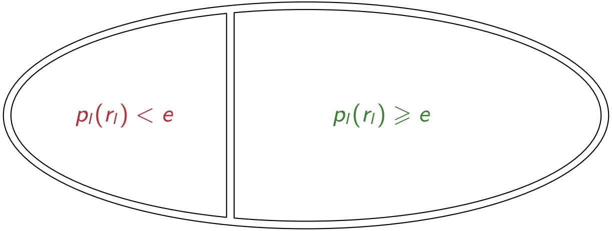
Let $\phi : (r_l, r_g) \longmapsto \phi(r_l, r_g)$ where r_g is the resampled value and r_l refers to other fixed samples.

We want to study the set $\left\{ \ r_l \ \middle| \ \phi(r_l, r_g) \text{ holds with non-negligible probability on } r_g \ \right\}$.

Let p_l be the following function

$$p_l \stackrel{\text{def}}{=} r_l \longmapsto \Pr_{r_g}\Big(\phi(r_l, r_g)\Big)$$

Sampling space (on r_l)



Another addition to the CCSA logic

Selection of sampling space predicate

Let $\phi : (r_l, r_g) \longmapsto \phi(r_l, r_g)$ be a function predicate.

Variable r_g is the parameters we want to rewind in the predicate ϕ.

select-tape is a local predicate saying that locally we are in the "good" case where ϕ holds.

select-tape predicate

$$[\![\textbf{select-tape}(e, \phi(r_l))]\!]_\rho \stackrel{\text{def}}{=} \Pr_{r_g}\left([\![\phi(r_l)]\!]_\rho(r_g)\right) \geqslant e.$$

Proof strategy - Step 1

Goal proof under select-tape guard - Axiomatization

The G.EXTRACT rule becomes

G.SEL-INTRO
$$[\textbf{select-tape}(e, \psi_{\mathcal{R}}(r_l)) \rightarrow (x(r_l), \textbf{zkp-extract}_{\mathcal{R}}(x(r_l), \mathfrak{p}^{(1)}_{\mathcal{R}}(r_l, \textbf{resample}(r_g)), \mathfrak{p}^{(2)}_{\mathcal{R}}(r_l, \textbf{resample}(r_g))))]$$

Where $\psi_{\mathcal{R}}(r_l) \stackrel{\text{def}}{=} r_g \longmapsto \textbf{zkp-verif}_{\mathcal{R}}(x(r_l), (p^0_{\mathcal{R}}(r_l), r_g, p^1_{\mathcal{R}}(r_g)))$.

Rewinding lemma

Statement

resample predicate

Let $\phi : r_g \longmapsto \phi(r_g)$ be a predicate. If $\mathbf{r}_g : \mathbf{nat} \to \tau_g$ then

$$\tilde{\exists}\, k : \mathbf{nat}\ [\text{poly}].\ \tilde{\exists}\, \mathbf{resample} : \mathbf{list}_n(\tau_g) \to \tau_g.$$
$$[\mathbf{select\text{-}tape}(e, \phi) \to \phi(\mathbf{resample}(\mathbf{r}_g\, 1, \ldots, \mathbf{r}_g\, k))]$$

Proof strategy - Step 2

Glue splitted parts back together

$\mathcal{H} : r \longmapsto \mathcal{H}\ r$ (Hypothesis predicate); Goal : $r \longmapsto$ Goal r (Goal predicate).

$$\dfrac{\textsc{G.Sel-Elim} \quad \tilde{\forall} e : \textbf{real}\ [\text{non-negl}].[\mathbf{select\text{-}tape}(e, \mathcal{H}) \rightarrow \mathcal{H}\ r \rightarrow \text{Goal}\ r]}{[\mathcal{H}\ r \rightarrow \text{Goal}\ r]}$$

Proof strategy - Step 2

Glue splitted parts back together

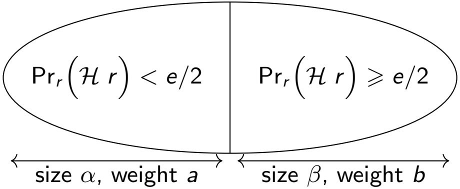
$\mathcal{H} : r \longmapsto \mathcal{H}\ r$ (Hypothesis predicate); Goal : $r \longmapsto$ Goal r (Goal predicate).

$$\frac{\tilde{\forall} e : \textbf{real}\ [\text{non-negl}].[\textbf{select-tape}(e, \mathcal{H}) \rightarrow \mathcal{H}\ r \rightarrow \text{Goal}\ r]}{[\mathcal{H}\ r \rightarrow \text{Goal}\ r]}\ \textsc{G.Sel-Elim}$$

Why does it work?

Proof by contraposition: we want to prove

$$\frac{{}_{e}[\mathcal{H}\ r \wedge \neg\, \text{Goal}\ r]}{{}_{e/2}\left[\textbf{select-tape}\left(\frac{e}{2}, \mathcal{H}\right) \wedge \mathcal{H}\ r \wedge \neg\, \text{Goal}\ r\right]}$$



We have $a \leqslant e/2$ and $b \leqslant \beta$.
Therefore, as $a + b \geqslant e$, $\beta \geqslant e/2$

Conclusion

Take aways

- To axiomatize rewinding argument, we have to resample only a part of the random tape;
- We need to talk about formulas sometimes true;
- High-order logic was needed for the rewinding lemma!

Other works done

- Complete formal proof of the permutation secrecy property;
- First complete proof of Terelius & Wikström mixnet protocol.

Conclusion

Take aways

To axiomatize rewinding argument, we have to resample only a part of the random tape;
We need to talk about formulas sometimes true;
High-order logic was needed for the rewinding lemma!

Other works done

Complete formal proof of the permutation secrecy property;
First complete proof of Terelius & Wikström mixnet protocol.

What next?

Reprogrammable Random Oracle Model

Sigma-protocols → NIZK proof (Fiat-Shamir transform) ...

... Towards proof of in practice used mixnet protocols (CHVote and Belenios).

Conclusion

Take aways

- To axiomatize rewinding argument, we have to resample only a part of the random tape;
- We need to talk about formulas sometimes true;
- High-order logic was needed for the rewinding lemma!

Other works done

- Complete formal proof of the permutation secrecy property;
- First complete proof of Terelius & Wikström mixnet protocol.

What next?

- Reprogrammable Random Oracle Model
- Sigma-protocols $\rightarrow$ NIZK proof (Fiat-Shamir transform) ...
- ... Towards proof of in practice used mixnet protocols (CHVote and Belenios).

Thank you for your attention![1]

[1] Icons comes from the Flaticons website (https://www.flaticon.com/)