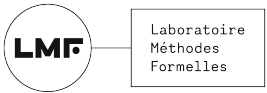
Mecanized proofs of electronic voting protocols: focus on mixnets protocols

Margot Catinaud

(1^{st}-year PhD Student)
Co-supervised by Caroline Fontaine & Guillaume Scerri

Séminaire au vert, June 2024



Electronic voting protocol

Voting protocol chronology

Voting phase

Tally

Electronic voting protocol

Voting protocol chronology

Voting phase

Tally

Main security properties looked for

Ballot privacy

Universal verifiability

Individual verifiability

Electronic voting protocol

Voting protocol chronology

Voting phase

Mixing phase

Tally

Main security properties looked for

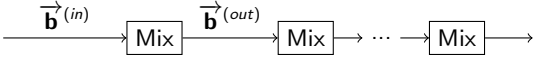
Ballot privacy

Universal verifiability

Individual verifiability

A key ingredient to ensure privacy: mix before tally

Principle



Network of *mix-servers*

Example: $\overrightarrow{\mathbf{b}}^{(in)} = \left(v_1\ {}_{r_1}, v_2\ {}_{r_2}, v_3\ {}_{r_3} \right)$ becomes

$\overrightarrow{\mathbf{b}}^{(out)} = \left(v_3\ {}_{s_1}, v_1\ {}_{s_2}, v_2\ {}_{s_3} \right)$

Algorithm : Mixing

let `mixing` $\overrightarrow{\mathbf{b}}^{(in)}$ **=**

$\pi \xleftarrow{\$} \mathfrak{S}_N$;

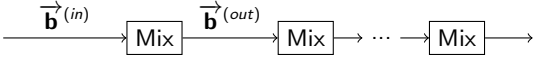
[*do some stuff...*] ;

return $\overrightarrow{\mathbf{b}}^{(out)}$

Mix-server in a nutshell

A key ingredient to ensure privacy: mix before tally

Principle



Network of *mix-servers*

Example: $\overrightarrow{\mathbf{b}}^{(in)} = \left(v_1\ {}_{r_1}, v_2\ {}_{r_2}, v_3\ {}_{r_3} \right)$ becomes

$\overrightarrow{\mathbf{b}}^{(out)} = \left(v_3\ {}_{s_1}, v_1\ {}_{s_2}, v_2\ {}_{s_3} \right)$

Algorithm : Mixing

let `mixing` $\overrightarrow{\mathbf{b}}^{(in)}$ **=**

$\pi \xleftarrow{\$} \mathfrak{S}_N$;

[*do some stuff...*] ;

return $\overrightarrow{\mathbf{b}}^{(out)}$

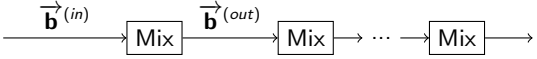
Mix-server in a nutshell

Desired security properties

- Ballot content stays **untouched**
- The attacker **can not** guess the permutation used to mix the ballot box

A key ingredient to ensure privacy: mix before tally

Principle



Network of *mix-servers*

Example: $\overrightarrow{\mathbf{b}}^{(in)} = \left(v_1\ {}_{r_1}, v_2\ {}_{r_2}, v_3\ {}_{r_3} \right)$ becomes

$\overrightarrow{\mathbf{b}}^{(out)} = \left(v_3\ {}_{s_1}, v_1\ {}_{s_2}, v_2\ {}_{s_3} \right)$

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

Desired security properties

- Ballot content stays **untouched**
- The attacker **can not** guess the permutation used to mix the ballot box

Needed ingredients

- Some *algebraic properties* involving ...
- ... *zero-knowledge proofs* on cryptographic constructions ...
- ... such as *commitment schemes*.

Zoom on one security property: mix-server verifiability

Cryptographic game — Mix-server verifiability.

Context

Adversarial mix-server

Honest verifier $\mathcal{V}$

Zoom on one security property: mix-server verifiability

Cryptographic game — Mix-server verifiability.

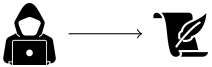
Context

Adversarial mix-server

Honest verifier $\mathcal{V}$

Game statement

Hypothesis



Proofs accepted by $\mathcal{V}$

$\Longrightarrow$

Conclusion

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

Equality of plaintexts lists as multisets

Sketch of proof: High level overview of reductions

Extraction of sealed matrix M

- *Witness extractor*
- *Collect enough witness*
- Reconstruction of sealed informations

Is M a permutation matrix?

- *Witness extractor*
- Witness consistency
- Generalization of equations on witness to equations on matrix
- Characterisation of permutation matrix

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

- *Another witness extractor*
- Consistency between the witness and the extracted matrix
- Generalization to the whole set of ciphertexts in/out pairs

Rewinding
Rewinding
Algebra

Rewinding
Cryptography
Algebra
Algebra

Rewinding
Cryptography
Algebra

The core idea [BC14]

Indistinguishability between terms

$$\boxed{t \sim u}$$

$$\Longleftrightarrow$$

With overwhelming probability, any attacker cannot distinguish terms t and u

A tool : SQUIRREL ([Bae+21])

Conclusion

What has been done?

- First complete formal proof of mix-server security properties in the CCSA model
- Filling gaps inside previous both cryptographic and formal proofs
- First formalization of the **rewinding** technique

What next?

- Σ-protocols $\longrightarrow$ NIZK proofs
- Security properties for mixnets
- Implement proofs in the SQUIRREL prover
- Some *modularity* in the proof

Conclusion

What has been done?

- First complete formal proof of mix-server security properties in the CCSA model
- Filling gaps inside previous both cryptographic and formal proofs
- First formalization of the **rewinding** technique

What next?

- Σ-protocols $\longrightarrow$ NIZK proofs
- Security properties for mixnets
- Implement proofs in the SQUIRREL prover
- Some *modularity* in the proof

Thank you for your attention!