A comprehensive analysis of Belenios

Alexandre Debant, and Véronique Cortier

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

GT-MFS

Fréjus, March 21st 2022
E-voting today

Hauts-de-Seine: Neuilly-sur-Seine met en place un système de vote électronique

La mairie de Neuilly-sur-Seine va tester un système de vote électronique pour permettre aux habitants d’arbitrer des décisions locales. Imaginée par l’association française.

Le vote électronique fera son retour en 2022

Après la découverte de failles en 2019, tous les projets de déploiement ont été suspendus. La Poste a cependant développé à Neuchâtel un système qui mettra à des hackers.
General information

- **developers:** Véronique Cortier, Pierrick Gaudry, Stéphane Glondu
- **context:** developed for associative or professional elections
- +2000 elections in 2021, +110,000 ballots
- multi-languages platform: French, English, Spanish…

Technical details

- re-vote
- homomorphic tally and/or mixnets
- threshold decryption ($k$ ou of $n$ decryption trustees)
- weighted votes
- …
Security properties

**Vote secrecy** - no one is able to learn who I voted for!

- I vote 0
- I vote 1

≈

- I vote 1
- I vote 0
Security properties

**Vote secrecy** - no one is able to learn who I voted for!

<table>
<thead>
<tr>
<th>Vote 0</th>
<th>Vote 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>🖍</td>
<td>🤘</td>
</tr>
</tbody>
</table>

≈

<table>
<thead>
<tr>
<th>Vote 1</th>
<th>Vote 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>🤘</td>
<td>🖍</td>
</tr>
</tbody>
</table>

**Verifiability** - no one is able to modify the result of an election!

- **Eligibility**: all the counted ballots belong to legitimate voters
- **Individual verifiability**: if I see my last ballot on the bulletin board, it will be counted
- **Universal verifiability**: the result corresponds to the content of the ballot box
A complex environment...

A lot of participants… with different roles...

- Registrar
- Public bulletin board
- Voting server

Decryption trustees

Voters
A complex environment...

A lot of participants… with different roles…

- Registrar
- Public bulletin board
- Voting server
- Decryption trustees
- Voters

And complex scenarios…

- Re-vote
- Two-round elections
- Multiple ballot-boxes (e.g., one per village/city)
Security claims

- verifiability as soon as the registrar or the voting server is honest
Security claims

- verifiability as soon as the registrar or the voting server is honest

- vote secrecy as soon as \( k \) out of \( n \) decryption trustees are honest
Security claims

- verifiability as soon as the registrar or the voting server is honest

ballot re-ordering attacks if re-vote is allowed

[Baloglu et al.- CSF’21]

- vote secrecy as soon as $k$ out of $n$ decryption trustees are honest
Ballot re-ordering attack
[Baloglu et. al.- CSF’21]

**Individual verifiability** - if I see my last ballot on the bulletin board, it will be counted

**Attack scenario**
Ballot re-ordering attack

[Baloglu et. al.- CSF’21]

**Individual verifiability** - if I see my last ballot on the bulletin board, it will be counted

**Attack scenario**
Ballot re-ordering attack

[Baloglu et. al.- CSF’21]

**Individual verifiability** - if I see my last ballot on the bulletin board, it will be counted

**Attack scenario**
Ballot re-ordering attack

[BAloglu et. al.- CSF’21]

**Individual verifiability** - if I see my last ballot on the bulletin board, it will be counted

**Attack scenario**
Ballot re-ordering attack

[Baloglu et. al.- CSF’21]

Individual verifiability - if I see my last ballot on the bulletin board, it will be counted

Attack scenario

The last ballot is Alice is not counted…
Fixing the attack with counters

1. The bulletin board is initialized with a counter set to 0
Fixing the attack with counters

1. The bulletin board is initialized with a counter set to 0

2. Alice adds the counter in her ballot
Fixing the attack with counters

1. The bulletin board is initialized with a counter set to 0

2. Alice adds the counter in her ballot

3. The server accepts ballots with a greater counter (compared to the last Alice’s accepted ballot) and increments it by 1
Fixing the attack with counters

1. The bulletin board is initialized with a counter set to 0

2. Alice adds the counter in her ballot

3. The server accepts ballots with a greater counter (compared to the last Alice’s accepted ballot) and increments it by 1

Security analysis in Proverif

- use of natural numbers natively supported by ProVerif since recently
- slightly simplify the use of counters in the model; prove the gap by hand
- rely on axioms/lemmas and [precise] to avoid inaccuracies of the tool
Security claims

- verifiability as soon as the registrar or the voting server is honest

**ballot re-ordering attacks if re-vote is allowed**

[Baloglu et al.- CSF’21]

- vote secrecy as soon as k out of n decryption trustees are honest
Security claims

- Verifiability as soon as the registrar or the voting server is honest.

- Ballot re-ordering attacks if re-vote is allowed [Baloglu et al.- CSF’21]

- Vote secrecy as soon as $k$ out of $n$ decryption trustees are honest.

- Multi-election attacks [NEW]
A privacy attack against Belenios

Election 1 (important election)

Election 2 (small/test election)
A privacy attack against Belenios

Election 1 (important election)

Election key = $pk_{el}$

$vk_A = pk(eid, cred_A)$

$vk_B = pk(eid, cred_B)$

Registrar

Election 2 (small/test election)
A privacy attack against Belenios

Election 1 (important election)
Election key = $p_{k_{el}}$

$v_{k_A} = pk(eid, cred_A)$
$v_{k_B} = pk(eid, cred_B)$

Registrar

Election 2 (small/test election)
Election key = $p_{k_{el}}$

The trustees can use the same keys
A privacy attack against Belenios

**Election 1 (important election)**

Election key = $pk_{el}$

$vk_A = pk(eid, cred_A)$
$vk_B = pk(eid, cred_B)$

**Election 2 (small/test election)**

Election key = $pk_{el}$

$vk_I$

$vk_A$
A privacy attack against Belenios

**Election 1 (important election)**

Election key = $pk_{el}$

$vk_A = pk(eid, cred_A)$

$vk_B = pk(eid, cred_B)$

**Registrar**

Election 2 (small/test election)

Election key = $pk_{el}$

$vk_I$ and $vk_A$

**Registrar**

Election 1: Only $vk_A$ and $vk_B$ are sent, but an attacker can intercept and see $vk_A$ and $vk_B$.

Election 2: $vk_I$ and $vk_A$ are also sent, allowing the registrar to authenticate the voter and ensure the integrity of the election.
A privacy attack against Belenios

Election 1 (important election)
Election key = $p_{ke}$

$v_k_A = pk(eid, cred_A)$
$v_k_B = pk(eid, cred_B)$

Registrar

The attacker can learn Alice’s vote

Election 2 (small/test election)
Election key = $p_{ke}$

Registrar

$(v_k_A, v_k_I)$

$v_k_A$
A privacy attack against Belenios

**Election 1 (important election)**

Election key = $p_{ke1}$

$vk_A = h(eid, cred_A)$

$vk_B = h(eid, cred_B)$

Registrar

**Election 2 (small/test election)**

Election key = $p_{ke1} p'_{ke1}$

$vk_A$

Registrar

(ballot rejected (invalid ZKP))

After v1.13 with an honest server

Fix - the server acts as a decryption trustee and must refresh its key for each election
Belenios - summary
[submitted at ESORICS’22]

Contributions:
- A (partial) fix: the Voting Server acts as a Trustee for decryption!
- A comprehensive model of Belenios including multi-elections
- Security proofs in ProVerif
- Paper proofs justifying the approximations about counters
# Belenios - summary

[submitted at ESORICS'22]

**Contributions:**
- A (partial) fix: the Voting Server acts as a Trustee for decryption!
- A comprehensive model of Belenios including multi-elections
- Security proofs in ProVerif
- Paper proofs justifying the approximations about counters

```
<table>
<thead>
<tr>
<th>Registrar</th>
<th>Server</th>
<th>Belenios &lt;v1.13</th>
<th>Belenios + Server Trustee</th>
<th>Belenios + Server Trustee + counters/commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verifiability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dis</td>
<td>Hon</td>
<td>✗ [CSF'21]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Hon</td>
<td>Dis</td>
<td>✗ [CSF'21]</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Privacy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dis</td>
<td>Hon</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hon</td>
<td>Dis</td>
<td>✗</td>
<td>✓ / ✗</td>
<td>✓ / ✗</td>
</tr>
</tbody>
</table>
```
Belenios - summary
[submitted at ESORICS’22]

Contributions:
- A (partial) fix: the Voting Server acts as a Trustee for decryption!
- A comprehensive model of Belenios including multi-elections
- Security proofs in ProVerif
- Paper proofs justifying the approximations about counters

<table>
<thead>
<tr>
<th></th>
<th>Registrar</th>
<th>Server</th>
<th>Belenios &lt;v1.13</th>
<th>Belenios + Server Trustee</th>
<th>Belenios + Server Trustee + counters/commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verifiability</td>
<td>Dis</td>
<td>Hon</td>
<td>✗ [CSF’21]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Hon</td>
<td>Dis</td>
<td>✗ [CSF’21]</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Privacy</td>
<td>Dis</td>
<td>Hon</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Hon</td>
<td>Dis</td>
<td>✗</td>
<td>✓ / ✗</td>
<td>✓ / ✗</td>
</tr>
</tbody>
</table>

Only if all the elections are audited
### Contributions:
- A (partial) fix: the Voting Server acts as a Trustee for decryption!
- A comprehensive model of Belenios including multi-elections
- Security proofs in ProVerif
- Paper proofs justifying the approximations about counters

### Table: Verifiability and Privacy

<table>
<thead>
<tr>
<th>Registrar</th>
<th>Server</th>
<th>Belenios &lt;v1.13</th>
<th>Belenios + Server Trustee</th>
<th>Belenios + Server Trustee + counters/commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dis</td>
<td>Hon</td>
<td>❌ [CSF’21]</td>
<td>❌</td>
<td>✅</td>
</tr>
<tr>
<td>Hon</td>
<td>Dis</td>
<td>❌ [CSF’21]</td>
<td>❌</td>
<td>✅</td>
</tr>
<tr>
<td>Dis</td>
<td>Hon</td>
<td>❌</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hon</td>
<td>Dis</td>
<td>❌</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

- **Based on counters to avoid ballot-reordering attacks**

- **Only if all the elections are audited**
Take home

Design and prove the security of an e-voting protocol is difficult…

- re-vote policies
- multiple elections
- …
Design and prove the security of an e-voting protocol is difficult…

- re-vote policies
- multiple elections
- …

The Swiss solution was also vulnerable to multi-elections attacks (will be presented at RWC’22)
Take home

Design and prove the security of an e-voting protocol is difficult...
- re-vote policies
- multiple elections
- ...

The Swiss solution was also vulnerable to multi-elections attacks (will be presented at RWC’22)

Main idea of the attack
Take home

Design and prove the security of an e-voting protocol is difficult…

- re-vote policies
- multiple elections
- …

The Swiss solution was also vulnerable to multi-elections attacks (will be presented at RWC’22)

Main idea of the attack

Decryption trustees

Learns Alice’s vote
Future works

- Ensure cast-as-intended in Belenios
  - model arithmetics
  - model probabilities (e.g., random audits)

- Study accountability (what happens in case of failure?)
  - design choice to improve Belenios
  - verify liveness properties