Fixing the Achilles Heel of E-Voting: The Bulletin Board

IEEE Computer Security Foundations Symposium 2021

Lucca Hirschi, Lara Schmid, David Basin

Inria          DFINITY          ETH Zurich
**Key goals**

1. **Vote privacy.** Threat model: 1 out of n tally servers 😇; other tally 😈, voting server 😈, BB 😈
Security proof of verifiability:

- Assume an idealized BB (sometimes implicit): ε

- Actual design, reference implementation, and deployments:
  - Voting server + BB → centralized server → single point of trust

Questions

1. How does BB impact verifiability in practice?
2. Which minimal BB requirement for Verifiability?
3. Is this requirement practical?

Contribution time!
Attack vector: BB equivocation

Equivocation: BB😈 shows a different content!
Main equivocation attack

涔(can induce a bias for涔:

1. **target a set of voters** who likely do **not** vote for
2. when casts a vote, **remove** from the BB, except for
3. proceed honestly with other voters and the auditors

→ Tally has #(target voters) less ballots against

Detection is overwhelmingly unlikely (more in the paper)
Other equivocation attacks

\(BB\) can equivocate on other data items towards different agents.

We found various such equivocation attacks on Civitas and Belenios/Helios:

<table>
<thead>
<tr>
<th>Threat Model</th>
<th>Violate</th>
<th>Equivocation (content, reader)</th>
<th>PD?</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 none (hon. tellers)</td>
<td>IV</td>
<td>possible candidates, voters</td>
<td>✓</td>
</tr>
<tr>
<td>C.2 none (hon. tellers)</td>
<td>IV, UV</td>
<td>(public) credentials, TTs</td>
<td>✓</td>
</tr>
<tr>
<td>C.3 tabulation tellers</td>
<td>IV, UV</td>
<td>ballots on final BB, voters</td>
<td>✗</td>
</tr>
<tr>
<td>C.4 none (hon. tellers)</td>
<td>IV, UV</td>
<td>blocks on final BB, final readers</td>
<td>✓</td>
</tr>
<tr>
<td>C.5 none (hon. tellers)</td>
<td>EV, CR</td>
<td>per-block credentials, TTs</td>
<td>✓</td>
</tr>
<tr>
<td>B.1 decryption trustees</td>
<td>IV, UV</td>
<td>ballots on final BB, voters</td>
<td>✗</td>
</tr>
<tr>
<td>B.2 none</td>
<td>IV</td>
<td>ballots on non-final BB, voters</td>
<td>✗</td>
</tr>
</tbody>
</table>

Practical Detection? i.e., easy fix? (other than a secure BB)
Fix the mismatch and the e-voting protocols

- Verifiability definitions consider $BB^{管局}$, we define $Verifiability^{管局}$ accounting for $BB^{管局}$

- New BB requirement: $FA$ that is
  - sufficient for verifiability:
    $(Verifiability^{管局} \land BB \vdash FA) \Rightarrow Verifiability^{管局}(BB)$
  - provably minimal

- New easily deployable BB protocol + machine-checked proof $BB^{管局} \vdash FA$

---

One can securely replace the insecure BB (1 server) by our secure BB protocol → effectively weaken trust assumptions:

$Verifiability^{管局} \rightarrow Verifiability^{管局}$
Conclusion

Contributions:

1. 💀 Practical attacks on Helios, Belenios, and Civitas
2. 📋 New BB requirement that is provably sufficient for verifiability
3. 📡 A BB protocol that can be used to weaken trust assumptions & prevent 😈

Future work:

1. 💀 Implement our attacks in the wild + user studies
2. 📋 Adapt Verifiability to the probabilistic setting (instead of possibilistic)
3. 📡 Explore other trade-off threat model versus deployment cost
Backup slides

Our BB protocol design:

Assuming $\gamma$ satisfies $\gamma > \frac{n-nh}{2}$ **versus** $\gamma > \frac{2n}{3}$ (BFT).
We were looking for minimal requirements for verifiability (no availability)
- Readers agree on final state
- Readers that read in between, can be sure that it will be included in the final state
- We were looking for minimal requirements for verifiability (no availability)
  - Readers agree on final state
  - Readers that read in between, can be sure that it will be included in the final state
BB protocol

- We were looking for minimal requirements for verifiability (no availability)
  - Readers agree on final state
  - Readers that read in between, can be sure that it will be included in the final state

Only accept if \( \gamma > \frac{n-n_{\text{h}}}{2} \) matching signatures
\( (n_{\text{h}} = \text{honest peers}) \)
Why distributed ledger are not a perfect BB

- **Permissionless:**
  - rely on **economic incentives** ⇒ hard to quantify in the case of elections
  - transaction **costs**
  - often centralized in practice due to **pools**

- **Permissioned ledgers:** few distinguished nodes establish a consensus on data that can be publicly accessed by all other nodes
  - BFT, which requires **strictly stronger trust** assumptions than **our solution**