Koinonia: Verifiable E-Voting with Long-term Privacy

Huangyi Ge
Purdue University

Joint work with Sze Yiu Chau, Victor E Gonsalves, Huian Li, Tianhao Wang, Xukai Zou, Ninghui Li
Encryption-based E-Voting System

Voters
- Receive public key for voting
- Generate a ballot under encryption and send to the server

Bulletin Board
- Publish Voter ID, Encrypted Ballot

Trust Centralized Server

Ensuring Privacy

Approach 1: Use shuffling/mixing
- First shuffle the ballot, then decrypt the ballots
- Publish a ZK proof of shuffling correctness
- Can use multiple shuffling servers.

Approach 2: Use homomorphic encryption
- “Add up” all ballots, then decrypt
- Can use threshold crypto.

Weakness:
- Encrypted ballots may be decrypted in future.
Secret-Sharing-based E-Voting System

Teller 1 → ballot share 1
Teller 2 → ballot share 2
Teller 3 → ballot share 3
Teller 4 → ballot share 4

Voter

Bulletin Board
System Architecture in Koinonia

- **Election Authority (EA)**: Voter registration and publish election information.
- **Election Service Provider (ESP)**: Run E-Voting Website, provide **Bulletin Board**.
- **Tabulation Tellers**: Privacy: Uses additive secret sharing. Trust model: At least one teller is honest.

**Tellers**
- Teller 1
- Teller 2
- Teller 3
- Teller 4

**Clouds**
- Election Authority
- Voter
- ESP
Additive Secret Sharing for Privacy

Example of 5 Voters and 4 Tellers

<table>
<thead>
<tr>
<th>Voter</th>
<th>Vote C_i</th>
<th>Ballot Shares</th>
</tr>
</thead>
<tbody>
<tr>
<td>V_1</td>
<td>1</td>
<td>T_1</td>
</tr>
<tr>
<td>V_2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>V_3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>V_4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>V_5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Bulletin Board</td>
<td>3</td>
<td>Agg_1</td>
</tr>
</tbody>
</table>

Voter's votes

Received ballot shares of T_j

Voter V_i split votes and sends each ballot share to related Teller T_j

Voter V_i sends Ballot share V_{ij} to Teller T_j

Aggregates of T_j
Tallying on Koinonia

- Teller 1
- Teller 2
- Teller 3
- Teller 4
- Agg 1
- Agg 2
- Agg 3
- Agg 4

Voter

Bulletin Board
### Tallying on Koinonia

**Example of 5 Voters and 4 Tellers**

<table>
<thead>
<tr>
<th>Voter</th>
<th>Vote</th>
<th>Ballot Shares</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$T_1$</td>
</tr>
<tr>
<td>$V_1$</td>
<td>1</td>
<td>$V_{1,1}$</td>
</tr>
<tr>
<td>$V_2$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$V_3$</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$V_4$</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$V_5$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Bulletin Board</td>
<td>3</td>
<td>$\text{Agg}_1$</td>
</tr>
</tbody>
</table>

#### Outcome

Compute the sum of Aggregates
# Integrity Using Cryptographic Commitments

## Example of 5 Voters and 4 Tellers

<table>
<thead>
<tr>
<th>Voter</th>
<th>Vote</th>
<th>Ballot Shares</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>V₁</td>
<td>1</td>
<td>V₁₁</td>
<td>T₁</td>
<td>T₂</td>
<td>T₃</td>
<td>T₄</td>
</tr>
<tr>
<td>V₂</td>
<td>0</td>
<td>V₂₁</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V₃</td>
<td>1</td>
<td>V₃₁</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V₄</td>
<td>1</td>
<td>V₄₁</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V₅</td>
<td>0</td>
<td>V₅₁</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bulletin Board</td>
<td>3</td>
<td>Agg₁</td>
<td>Agg₂</td>
<td>Agg₃</td>
<td>Agg₄</td>
<td></td>
</tr>
</tbody>
</table>

### Commit of Share
- Included in the Ballot
- Unconditional hiding
- Computational binding
- The Pedersen Commitment Scheme

![Verifiable Outcome](image)
## Well-formed Ballot of Koinonia

Example of 5 Voters and 4 Candidates

<table>
<thead>
<tr>
<th>Voter</th>
<th>Vote Sum</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>V₁</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V₂</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V₃</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>V₄</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V₅</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bulletin Board</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Well-formed Ballot**
  - Witness-Indistinguishable Proof (WIP)
  - Reveal no Information about the Vote Shares
  - WIP Included in the Ballot

- **Published Final Outcome**
Publishing on Koinonia

- Received Ballots
- Final Outcome
- Can be Verified by the Third Party

Voter → Bulletin Board Publishing

Teller 1 → Teller 2 → Teller 3 → Teller 4

bulletin board
Adversary of Koinonia

One Teller is honest.

Teller 1

Teller 2

Teller 3

Teller 4

Local Observer

- See Ballot Shares in Plaintext on Controlled Parties
- Privacy Protection
  - Information Theoretical Hiding
  - Additive Secret Sharing

Voter

Bulletin Board
Adversary of Koinonia

Global Observer
- See Network Transcript
- Government Agency
  - PRISM program by NSA
- Store the Transcript, Decrypt in the Future

Future Adversary
- **Future** Privacy is Important.
  - One Teller is Honest.
- Outcome Integrity is Timely.
  - Important for **Current, but not for Future**

Quantum Computer
- Break Public Key Crypto
- But not Quantum-Safe Crypto

Quantum-Safe Communication

Voter

Teller 1

Teller 2

Teller 3

Teller 4

Bulletin Board
Unbounded Adversary

- Quantum-Safe Crypto is **Broken**

Local *Unbounded Observer*
- Privacy Protection
  - Additive Secret Sharing
  - Information Theoretical Hiding

Voter

Bulletin Board

One Teller is honest.
Unbounded Adversary

Global Unbounded Observer
- Impossible to Defense
- Network Transcript can be decrypted
Framework of Adversary Models

Computational capabilities

Current
Future
Unbounded

Local Observer

Global Observer

Future Privacy

<Global, Unbounded>

<Global, Future>

<Local, Unbounded>

<Global, Current>

Current Integrity

<Local, Future>

<Local, Current>

Impossible to Defense
Other Security Considerations

- Teller Deny of Service (DoS)
- Teller manipulating ballot shares
- Missing Ballot Attack
- Ballot Stuffing Attack
Implementation

• Koinonia system
  – Node.js

• Koinonia Libraries
  – Share and Ballot Generation, Verification Functions
  – Client: SJCL (Stanford Javascript Crypto Library)
  – Server: Native Code Optimization
    • Node.js C++ Addons

• Secure Communication and Future Privacy
  – Open Quantum Safe (OQS) with Stunnel\(^2\)

Performance

- One position, two candidates, and three Tellers
- 8 core i7-3770 3.40 GHz CPU, 16GB Ram

<table>
<thead>
<tr>
<th>Role</th>
<th>Time (ms ±%)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voter</td>
<td>301 ± 4.9%</td>
<td>Construct shares and ballot</td>
</tr>
<tr>
<td>Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teller</td>
<td>2.37 ± 22%</td>
<td>Accept a share</td>
</tr>
<tr>
<td>ESP</td>
<td>5.77 ± 27%</td>
<td>Accept a ballot</td>
</tr>
<tr>
<td>Verifier</td>
<td>11s</td>
<td>10,000 Ballots, 8 threads</td>
</tr>
</tbody>
</table>
Conclusion

• Koinonia
  – Current integrity and Future privacy
  – Additive secret sharing, Pedersen commitment, and WIP

• Open source
  – Light weighted
  – https://github.com/gehuangyi20/Koinonia
Q&A

Thank You

Huangyi Ge

geh@purdue.edu