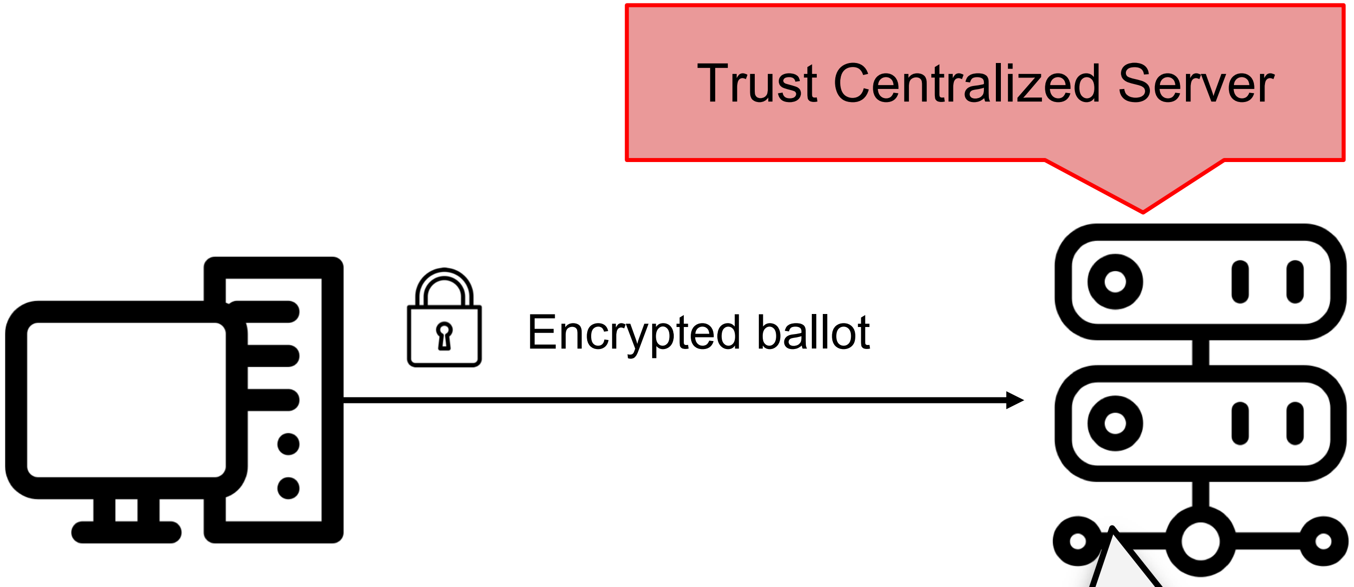
Koinonia: Verifiable E-Voting with Long-term Privacy

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



Bulletin Board

Publish Voter ID,
Encrypted Ballot

Voters

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

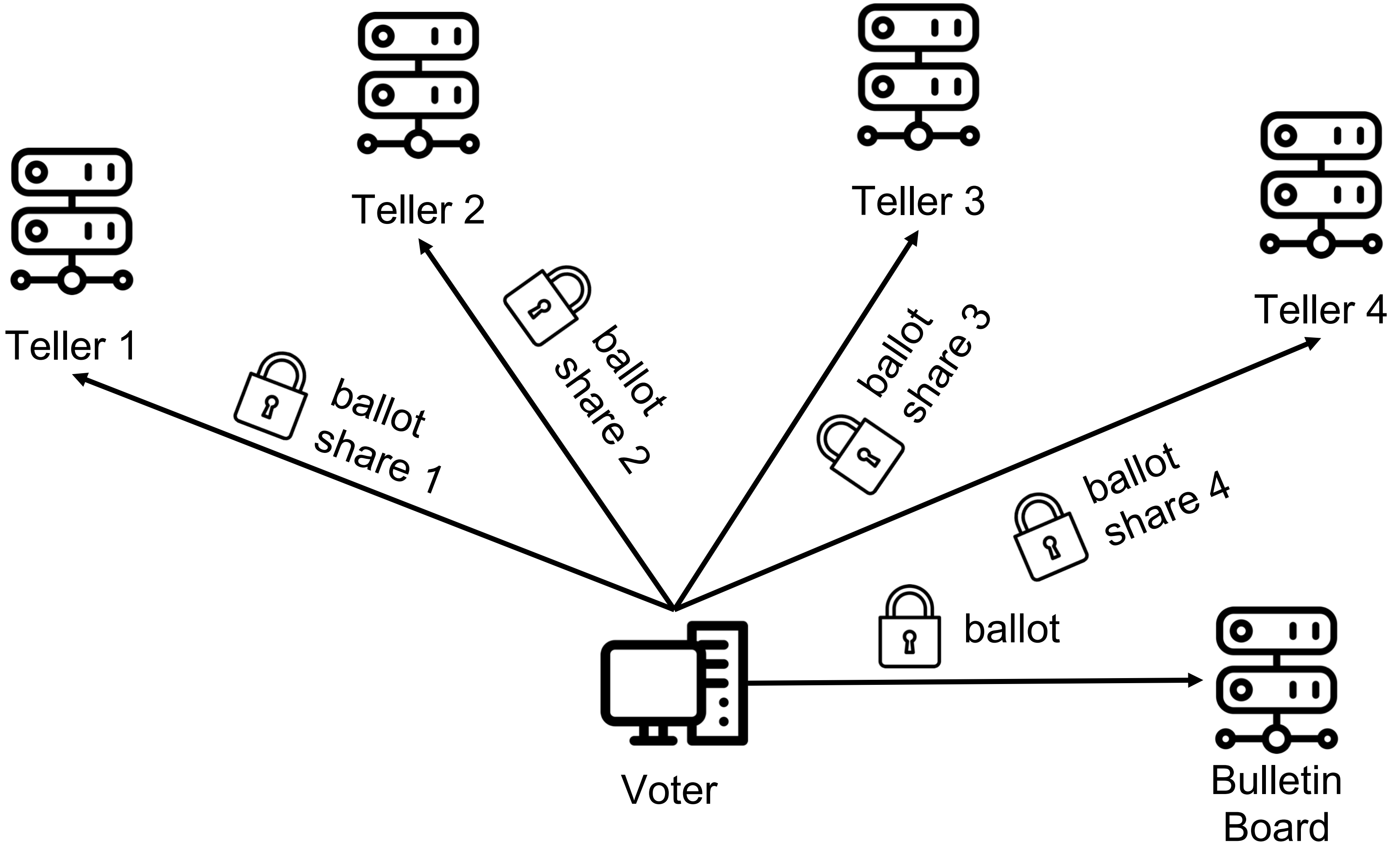
Approach 2: Use homomorphic encryption

- "Add up" all ballots, then decrypt
- Can use threshold crypto.

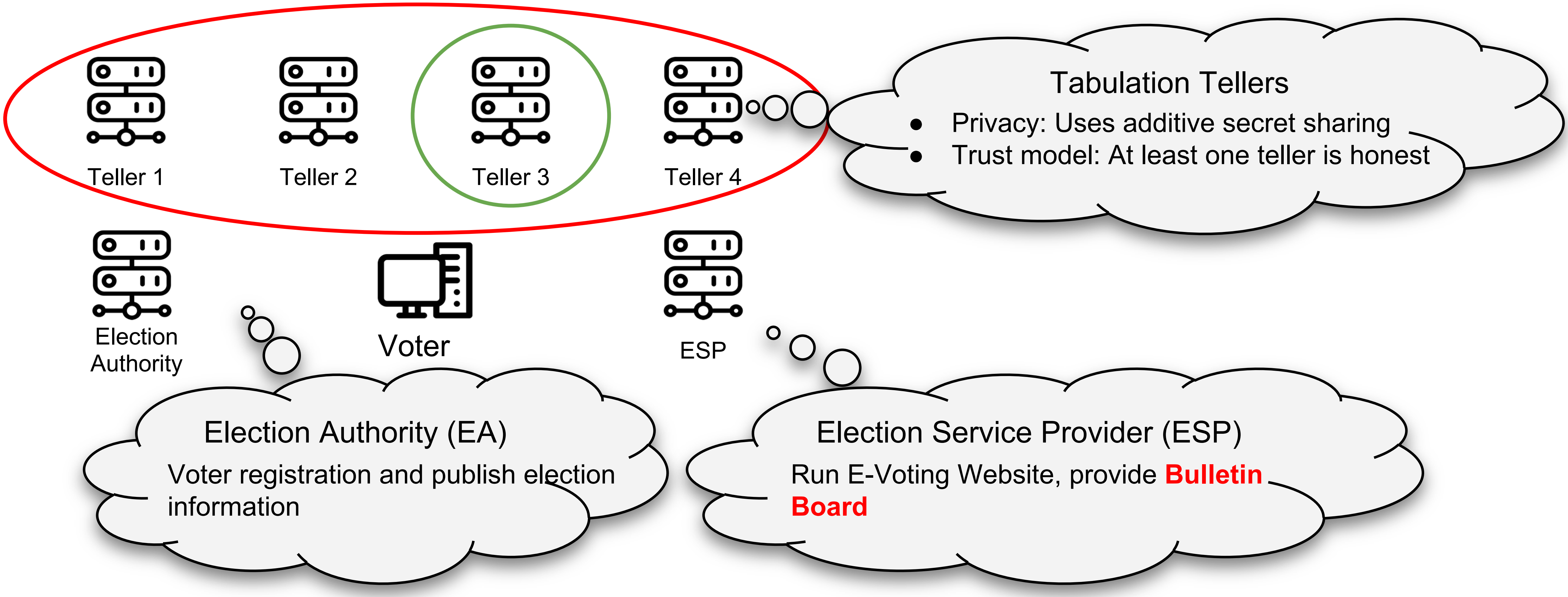
Weakness:

- Encrypted ballots may be decrypted in future.

Secret-Sharing-based E-Voting System



System Architecture in Koinonia



Additive Secret Sharing for Privacy

Voter's votes

Received ballot shares of T_j

Voter	Vote C_1	Ballot Shares			
		T_1	T_2	T_3	T_4
V_1	1	$V_{1,1}$	$V_{1,2}$	$V_{1,3}$	$V_{1,4}$
V_2	0	$V_{2,1}$			
V_3	1	$V_{3,1}$			
V_4	1	$V_{4,1}$			
V_5	0	$V_{5,1}$			
Bulletin Board	3	Agg_1	Agg_2	Agg_3	Agg_4

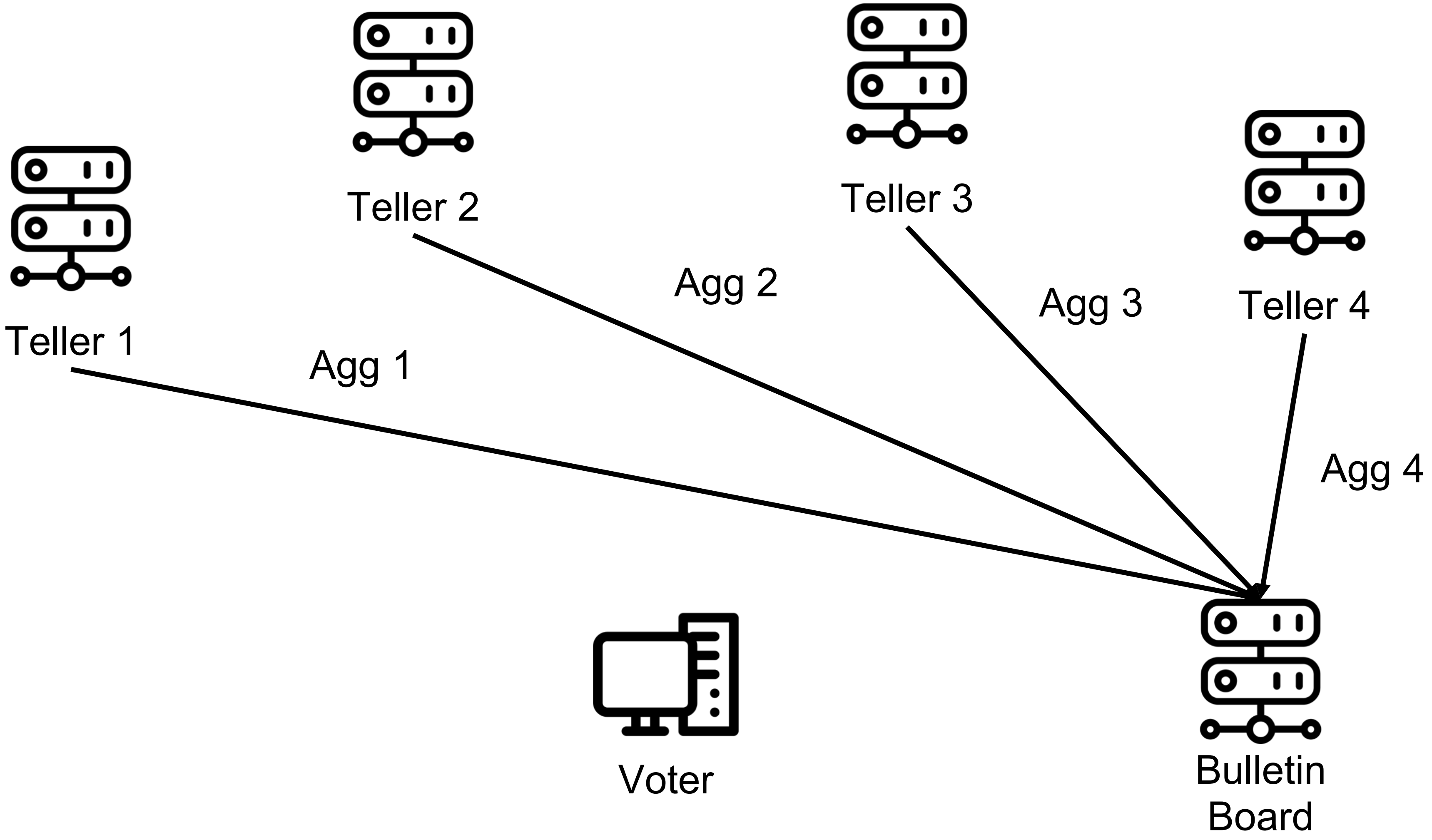
Voter V_i split votes and sends each ballot share to related Teller T_j

Voter V_i sends Ballot share $V_{i,j}$ to Teller T_j

Example of 5 Voters and 4 Tellers

Aggregates of T_j

Tallying on Koinonia



Tallying on Koinonia

Example of 5 Voters and 4 Tellers

Voter	Vote	Ballot Shares			
		T_1	T_2	T_3	T_4
V_1	1	$V_{1,1}$	$V_{1,2}$	$V_{1,3}$	$V_{1,4}$
V_2	0	$V_{2,1}$			
V_3	1	$V_{3,1}$			
V_4	1	$V_{4,1}$			
V_5	0	$V_{5,1}$			
Bulletin Board	3	Agg_1	Agg_2	Agg_3	Agg_4

Outcome

Compute the sum of Aggregates

Integrity Using Cryptographic Commitments

Example of 5 Voters and 4 Tellers

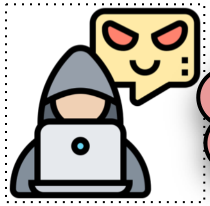
Voter	Vote	Ballot Shares			
		T_1	T_2	T_3	T_4
V_1	1	$V_{1,1}$	$V_{1,2}$	$V_{1,3}$	$V_{1,4}$
V_2	0	$V_{2,1}$			
V_3	1	$V_{3,1}$			
V_4	1	$V_{4,1}$			
V_5	0	$V_{5,1}$			
Bulletin Board	3	Agg_1	Agg_2	Agg_3	Agg_4

Verifiable Outcome

Commit of Share

- Included in the Ballot
- Unconditional hiding
- Computational binding
- The Pedersen Commitment Scheme

Well-formed Ballot of Koinonia



vote 2
or more

Example of 5 Voters and 4 Candidates

Voter	Vote Sum	Candidates			
		C_1	C_2	C_3	C_4
V_1	1	1	0	0	0
V_2	0	0	0	0	0
V_3	1	0	0	0	1
V_4	1	1	0	0	0
V_5	0	0	0	0	0
Bulletin Board	3	2	0	0	1

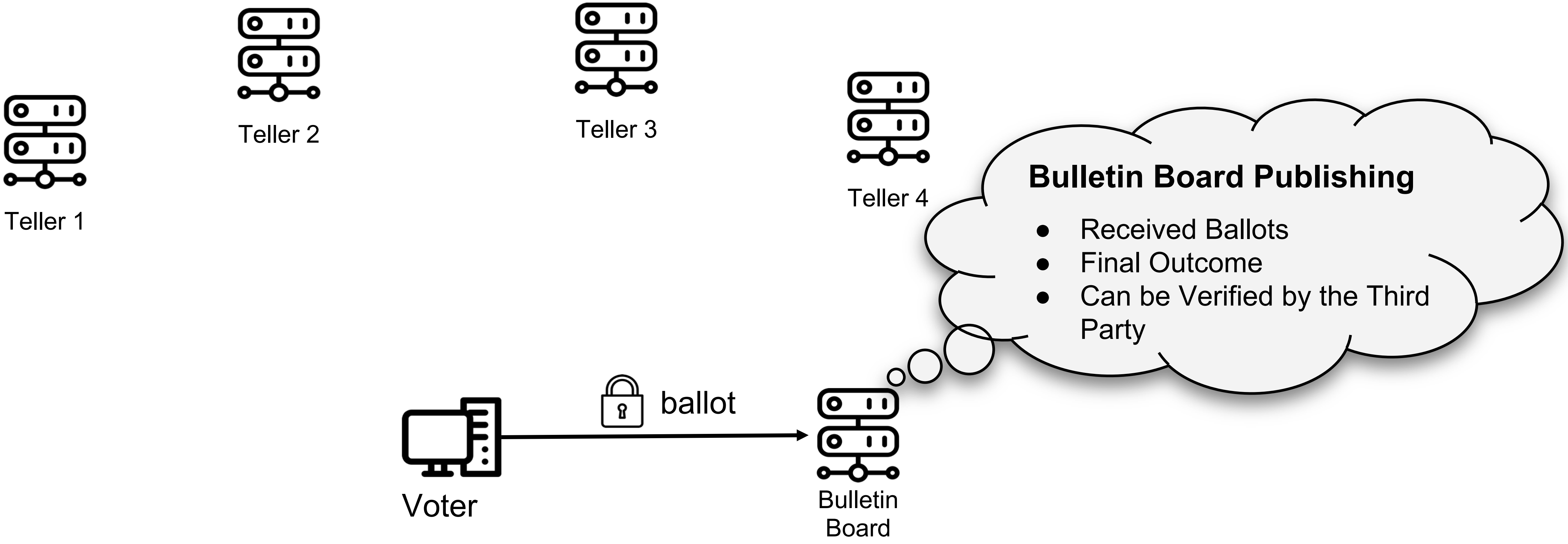
Ballot

Well-formed Ballot

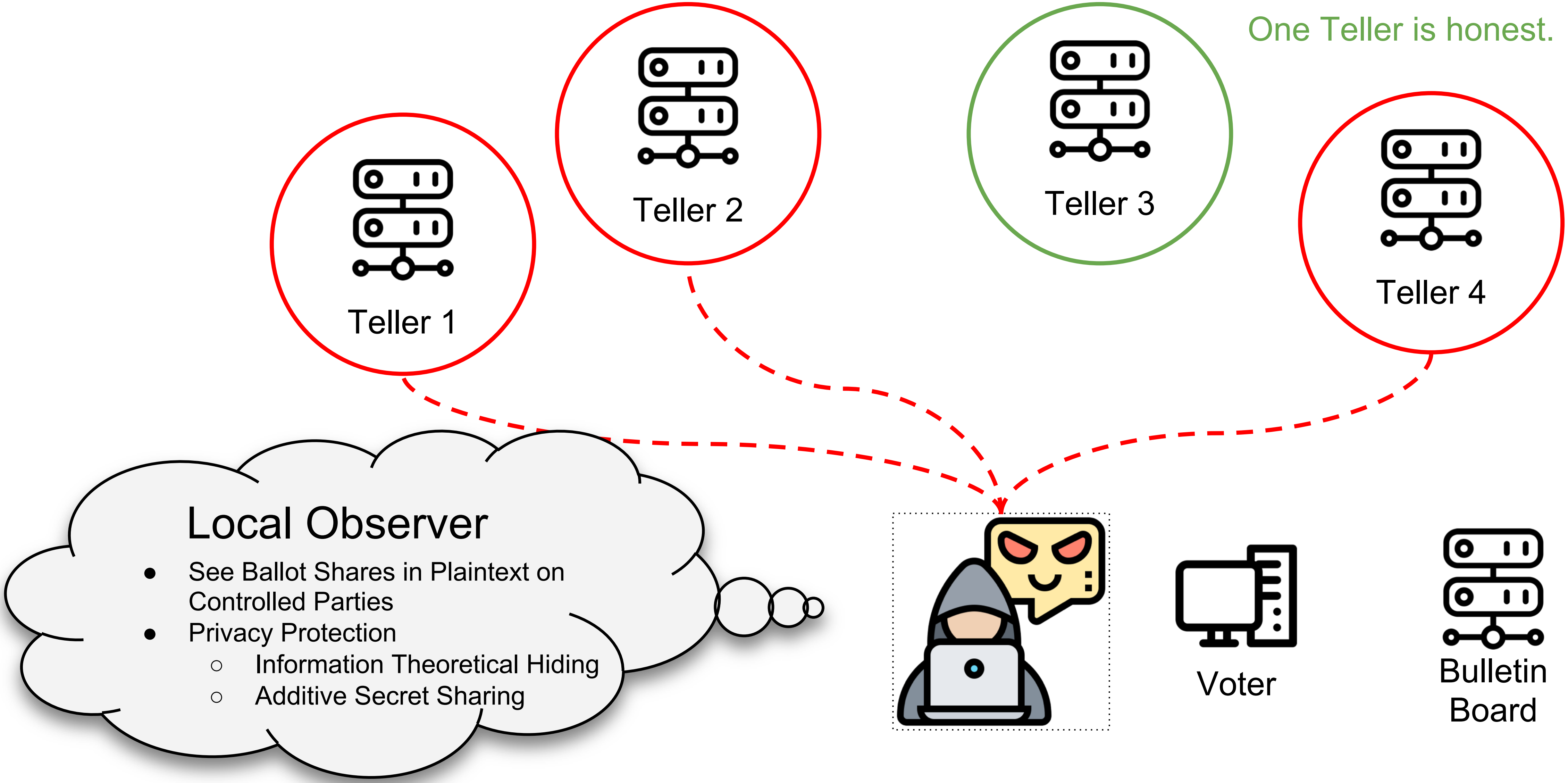
- Witness-Indistinguishable Proof (WIP)
- Reveal no Information about the Vote Shares
- WIP Included in the Ballot

Published Final Outcome

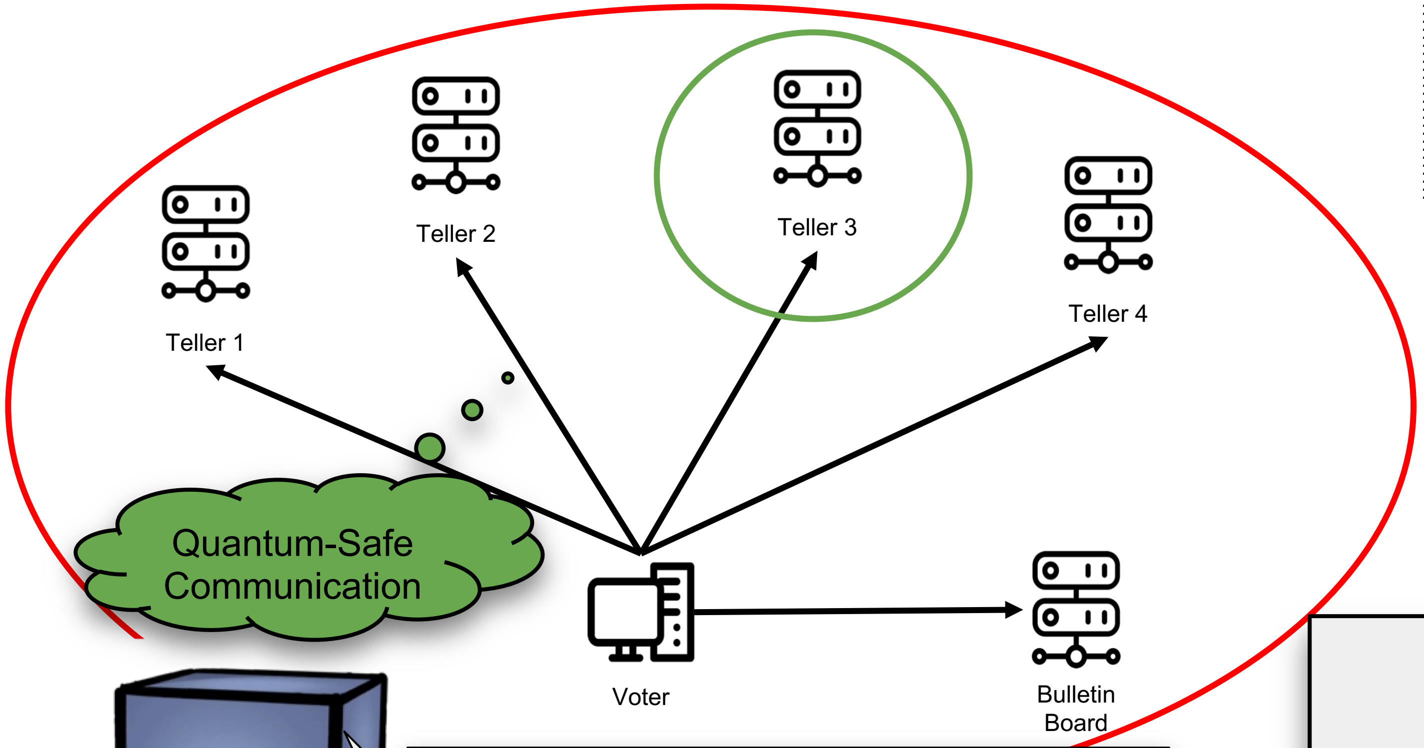
Publishing on Koinonia

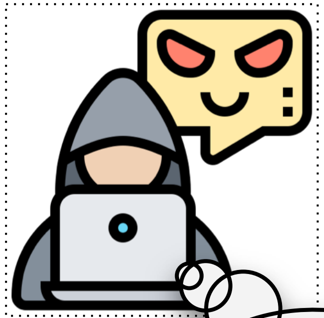


Adversary of Koinonia



Adversary of Koinonia



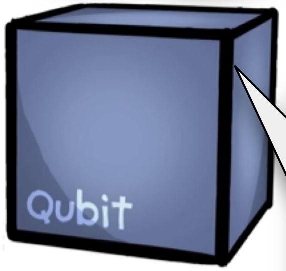


Global Observer

- 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

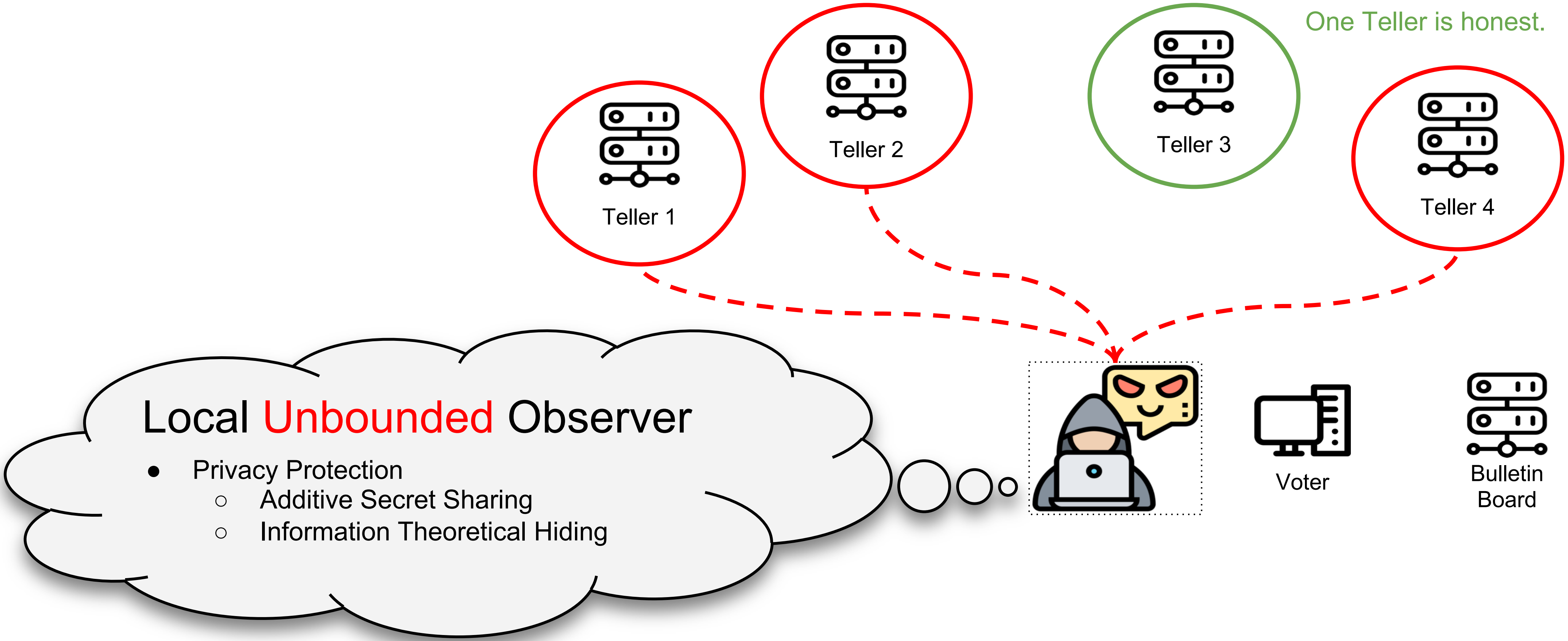


Quantum Computer

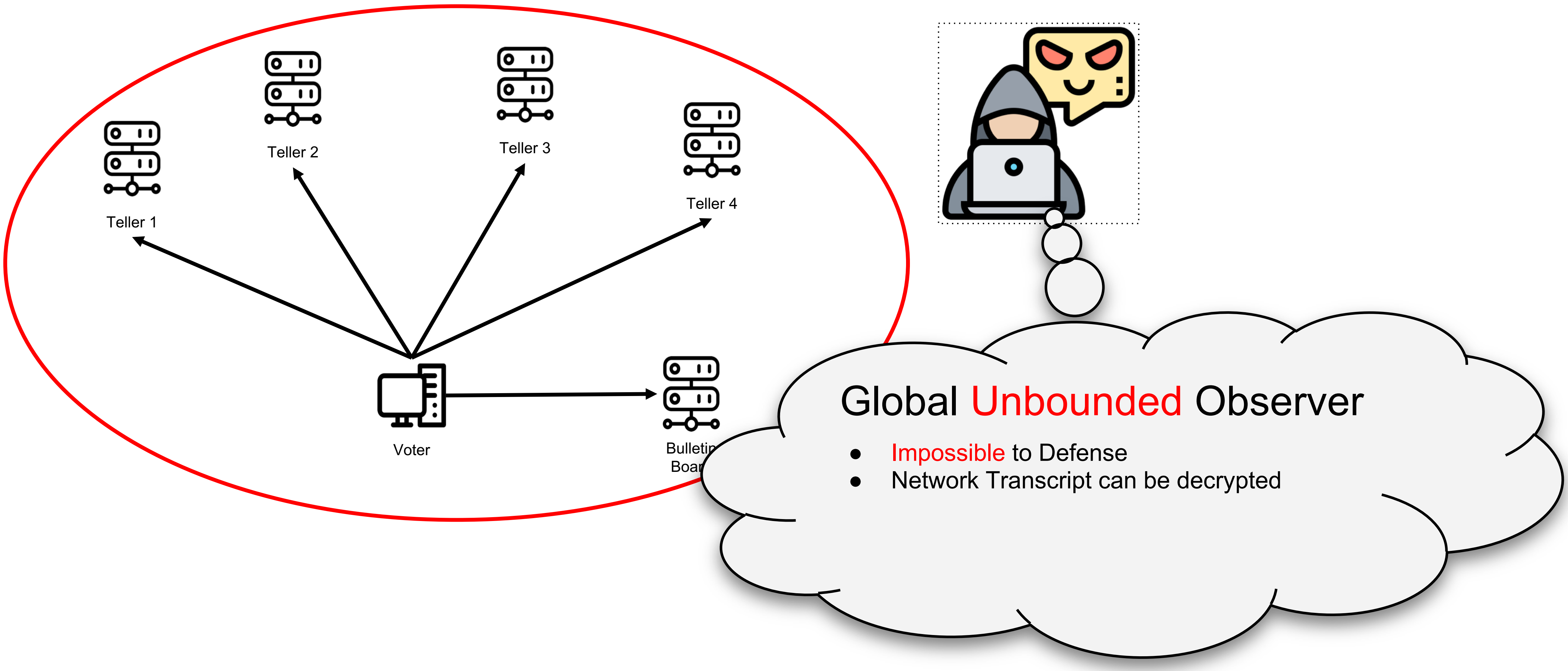
- Break Public Key Crypto
- But not Quantum-Safe Crypto

Unbounded Adversary

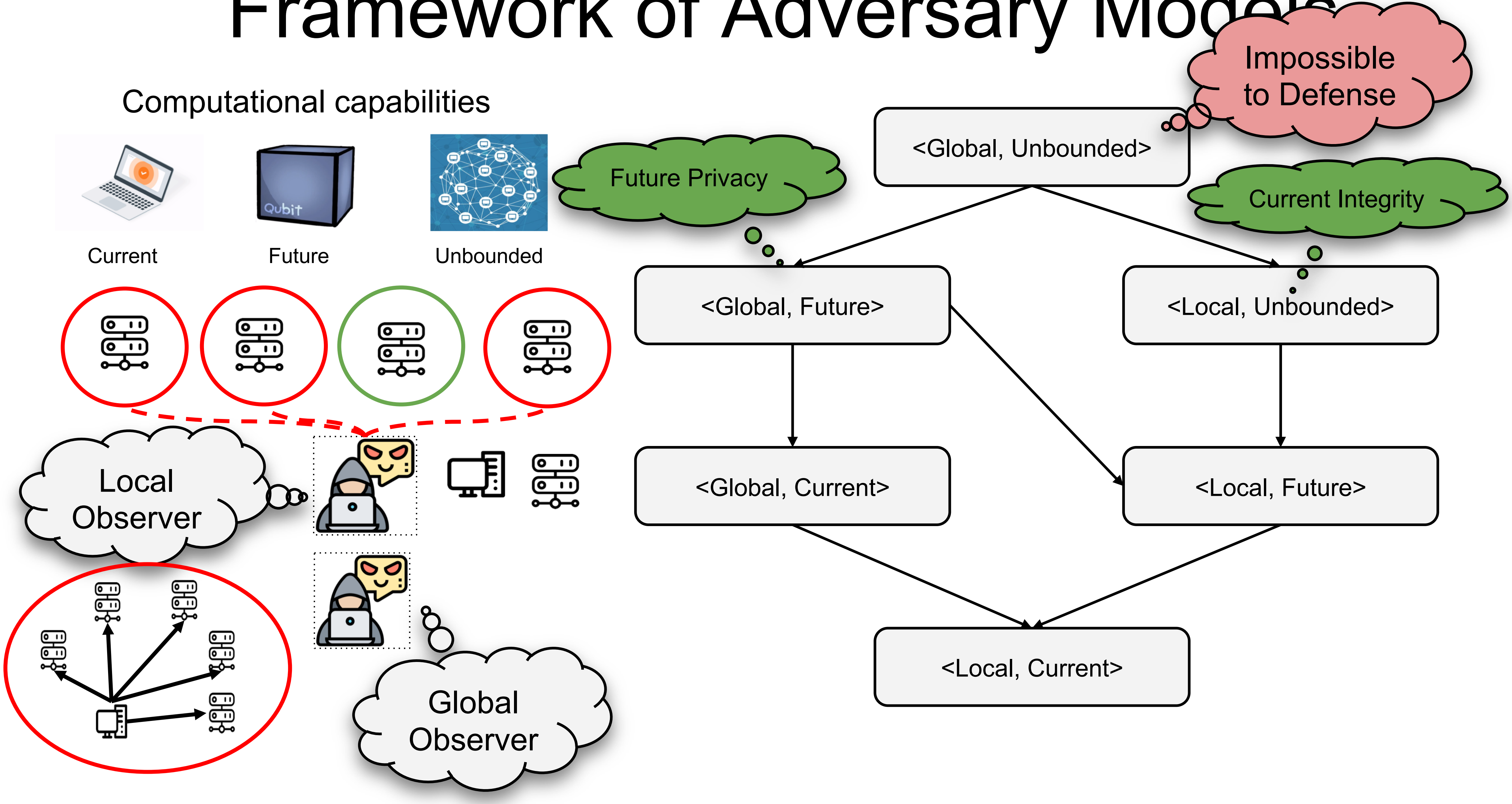
- Quantum-Safe Crypto is Broken



Unbounded Adversary



Framework of Adversary Models



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]

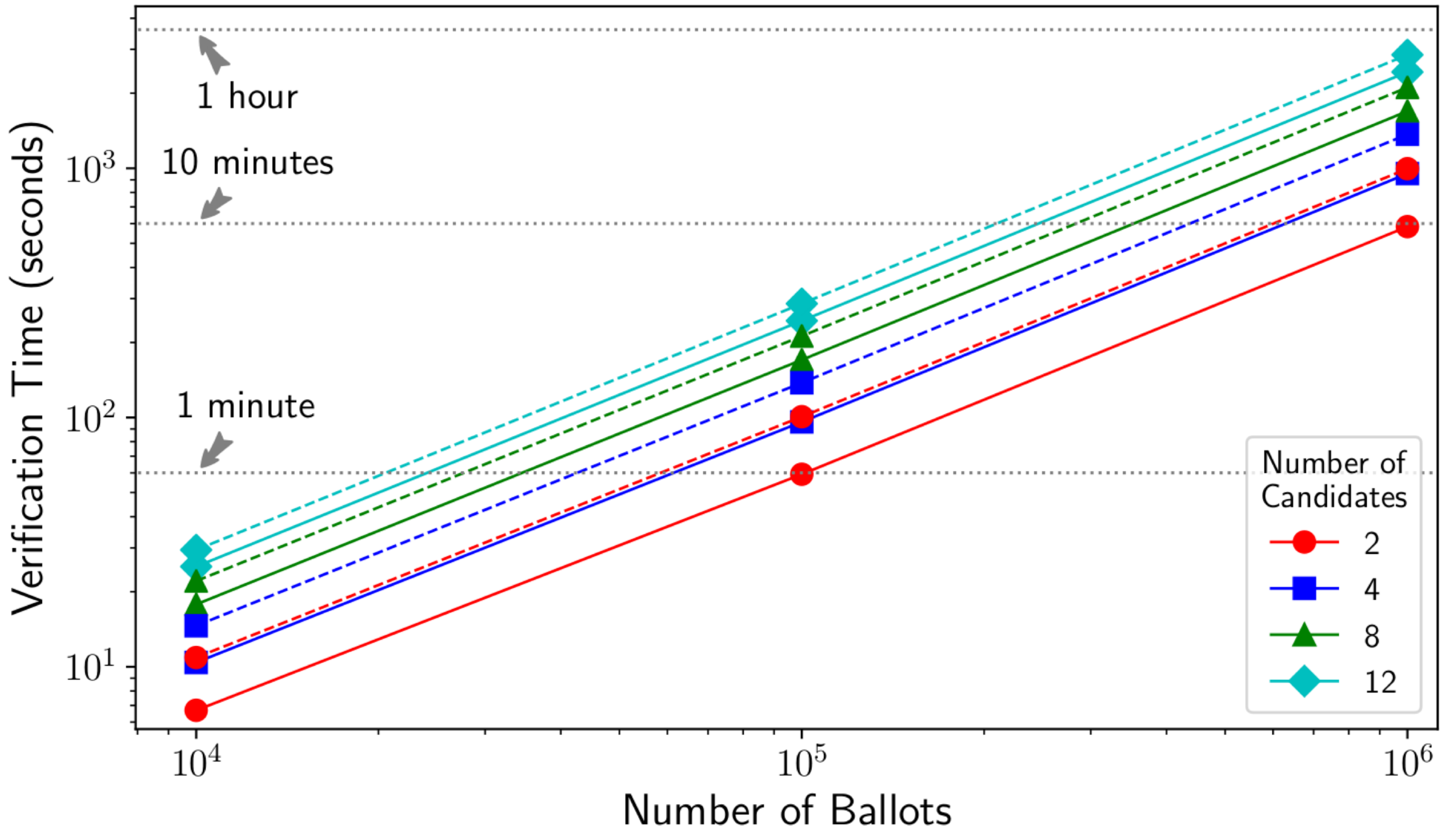
2. https://github.com/open-quantum-safe/liboqs

Performance

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

	Client	
Voter	301ms ± 4.9%	Construct shares and ballot
	Server	
Teller	2.37ms ± 22%	Accept a share
ESP	5.77ms ± 27%	Accept a ballot
Verifier	11s	10,000 Ballots, 8 threads

Verification Benchmark



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