Imperial College London

An End-to-End Verifiable Secure Online Voting System

Final Presentation

Author: Sander Sebastian Henschien Coates

Supervisor: Prof. William J. Knottenbelt

Co-supervisor: Clarissa Agnew

Date: August 2020

Agenda

1. Introduction
2. Background
3. Protocol Design
4. Security Analysis
5. Implementation
6. Evaluation
7. Conclusion

Introduction

Motivation

- Online voting systems have recently received remarkable attention
- Increased flexibility and participation
- Administrative cost savings
- Robustness against pandemics (e.g. COVID-19)
- Systems proposed so far have not earned sufficient public confidence
- Expectations of end-to-end verifiability, balanced with voter anonymity and usability, are challenging

The Iowa Caucus Tech Meltdown Is a Warning

MIT researchers identify security vulnerabilities in voting app

Mobile voting application could allow hackers to alter individual votes and may pose privacy issues for users.

POLITICS | ELECTION 2020

Democrats' Iowa Caucus Voting App Stirs Security Concerns

Caucus workers will use the app on their personal smartphones, prompting questions of possible vulnerability

American elections are too easy to hack. We must take action now

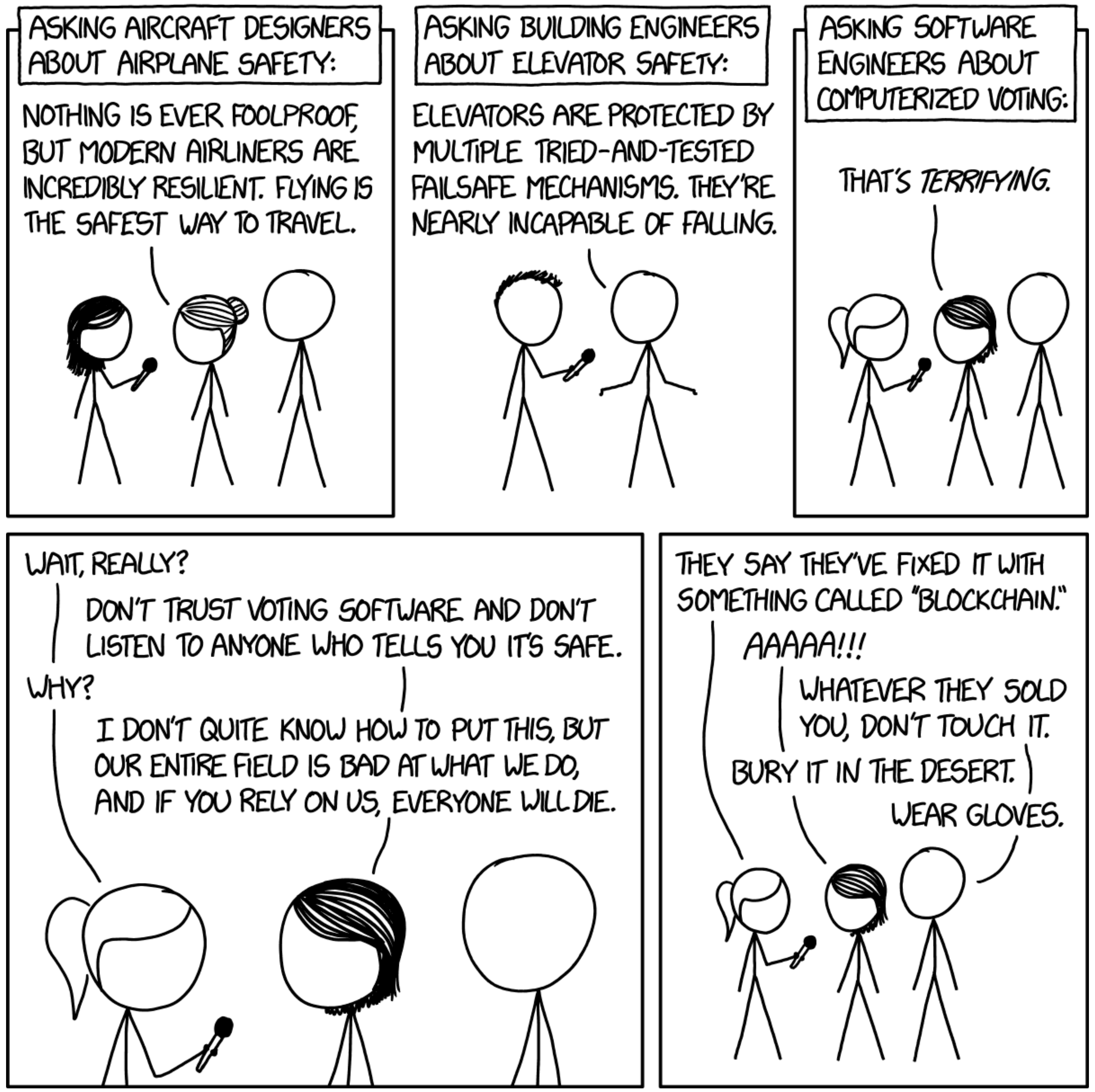
Newspaper headline references: [3-6]

Objectives

1. Design a new proof of concept secure online voting protocol that is practically feasible in terms of processing time and usability

2. Analyse the security of the protocol and propose measures to mitigate any potential threat

3. Demonstrate the protocol as implemented in a voting system and evaluate its practical feasibility both qualitatively and quantitatively



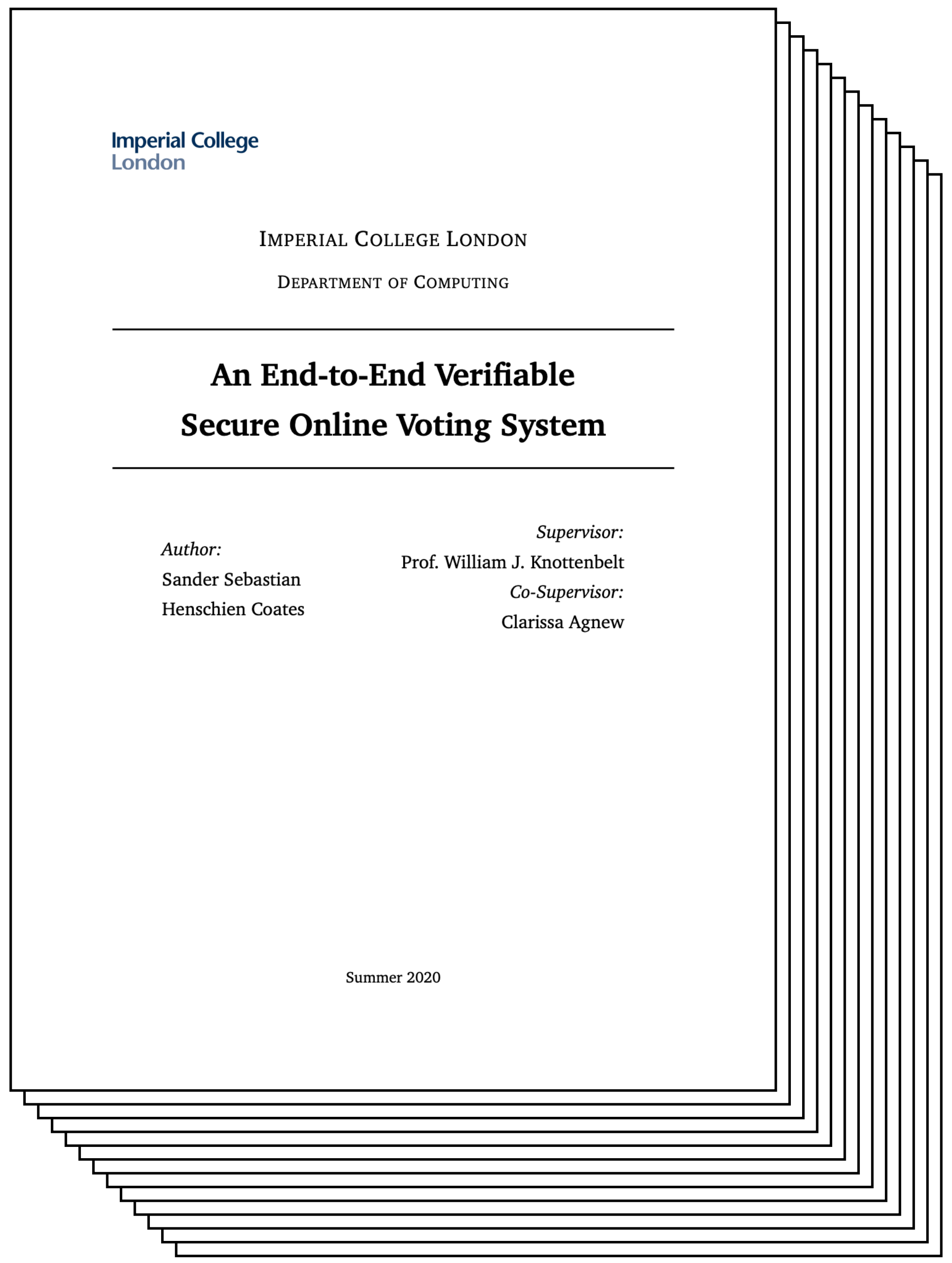
Comic available from https://xkcd.com/2030/
[cited 2020 Aug 20]

Contributions

- A new end-to-end verifiable secure online protocol inspired by the work of Ryan and Teague [1] and Chondros et al. [2] with improvements to robustness and usability

- Security analysis of the proposed protocol with suggested mitigating actions

- Demonstration of an end-to-end verifiable secure online system and evaluation of the protocol's practical feasibility


Imperial College
London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

An End-to-End Verifiable
Secure Online Voting System

Author:
Sander Sebastian
Henschien Coates

Supervisor:
Prof. William J. Knottenbelt
Co-Supervisor:
Clarissa Agnew

Summer 2020

Background

Technical preliminaries

- Vulnerabilities, exploits and attacks
- Threat modelling
- Malware
- Pseudo-randomness
- Cryptographic primitives
- Hashing and SHA
- Symmetric encryption and AES
- Asymmetric encryption and RSA
- Digital signatures
- Blockchain
- Homomorphic encryption
- Threshold encryption
- Mix networks
- Base64

Requirements for secure online voting systems

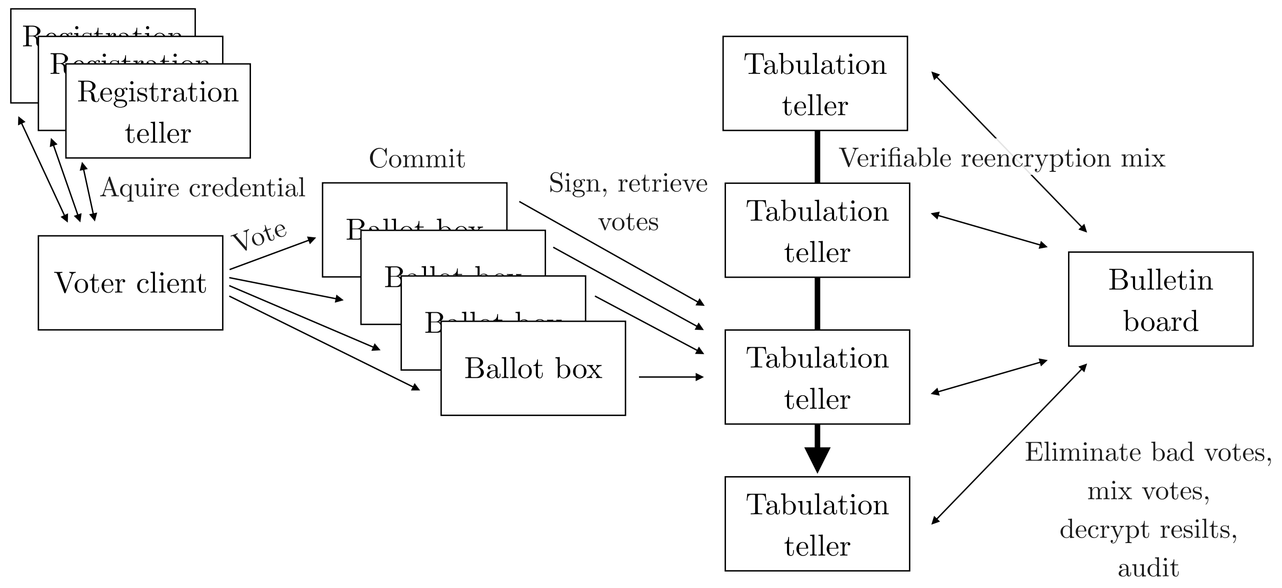
- End-to-end (E2E) verifiability
- Voter authentication and authorisation
- Receipt freedom
- Secret ballot
- Usability and accessibility
- Software independence
- Transparency
- Coercion resistance

Existing systems

- Rijnland Internet Election System (RIES)
- Civitas
- Pretty Good Democracy (PGD)
- Selene
- IVXV (Estonian I-Voting)
- Votem
- Voatz
- D-DEMOS

serial-no		
	Part A	
vote-code$_1$	option$_1$	vote-receipt$_1$
...	...	...
vote-code$_m$	option$_m$	vote-receipt$_m$
	Part B	
vote-code$_1$	option$_1$	vote-receipt$_1$
...	...	...
vote-code$_m$	option$_m$	vote-receipt$_m$

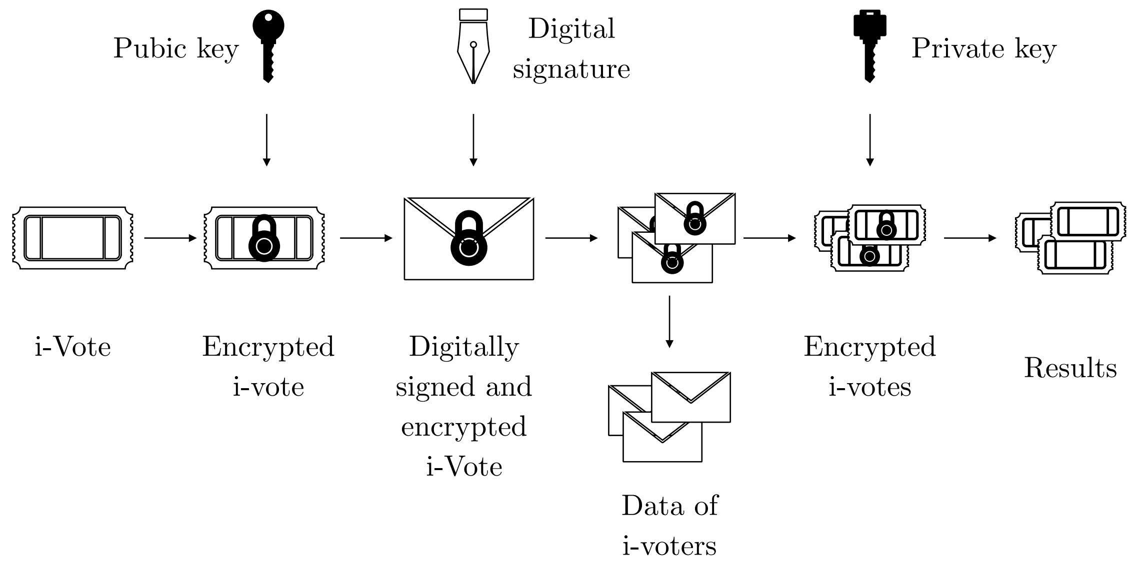
D-DEMOS



Civitas

Candidate	Vote code
Asterix	3772
Idefix	4909
Obelix	9521
Panoramix	7387
Ack Code: 8243	
Ballot Id: 3884092844	

PGD



Estonian I-Voting

Figure references: [1, 2, 34, 40]

Main limitations of existing systems

- Secret ballots and receipt freedom
- Coercion resistance
- Balance between anonymity and E2E verifiability
- Client side vulnerabilities
- Lacking usability

Requirement \ System	[39]	[34]	[1]	[32]	[40]*	[10]	[41]	[2]
E2E verifiability	✓	✓	✓	✓	✓	✓	-	✓
Voter authorisation and authentication	✓	✓	-	✓	✓	✓	✓	✓
Receipt freedom	-	✓	-	-	✓	✓	✓	✓
Secret ballot	-	✓	✓	✓	✓	✓	-	✓
Usability and accessibility	-	-	✓	✓	✓	✓	✓	✓
Software independence	✓	✓	✓	✓	✓	✓	-	✓
Transparency	✓	✓	✓	✓	✓	-	-	✓
Coercion resistance	-	✓	-	✓	✓	✓	✓	-

*Although doing seemingly good with regards to the requirements, Springall et al. [31] find several vulnerabilities.

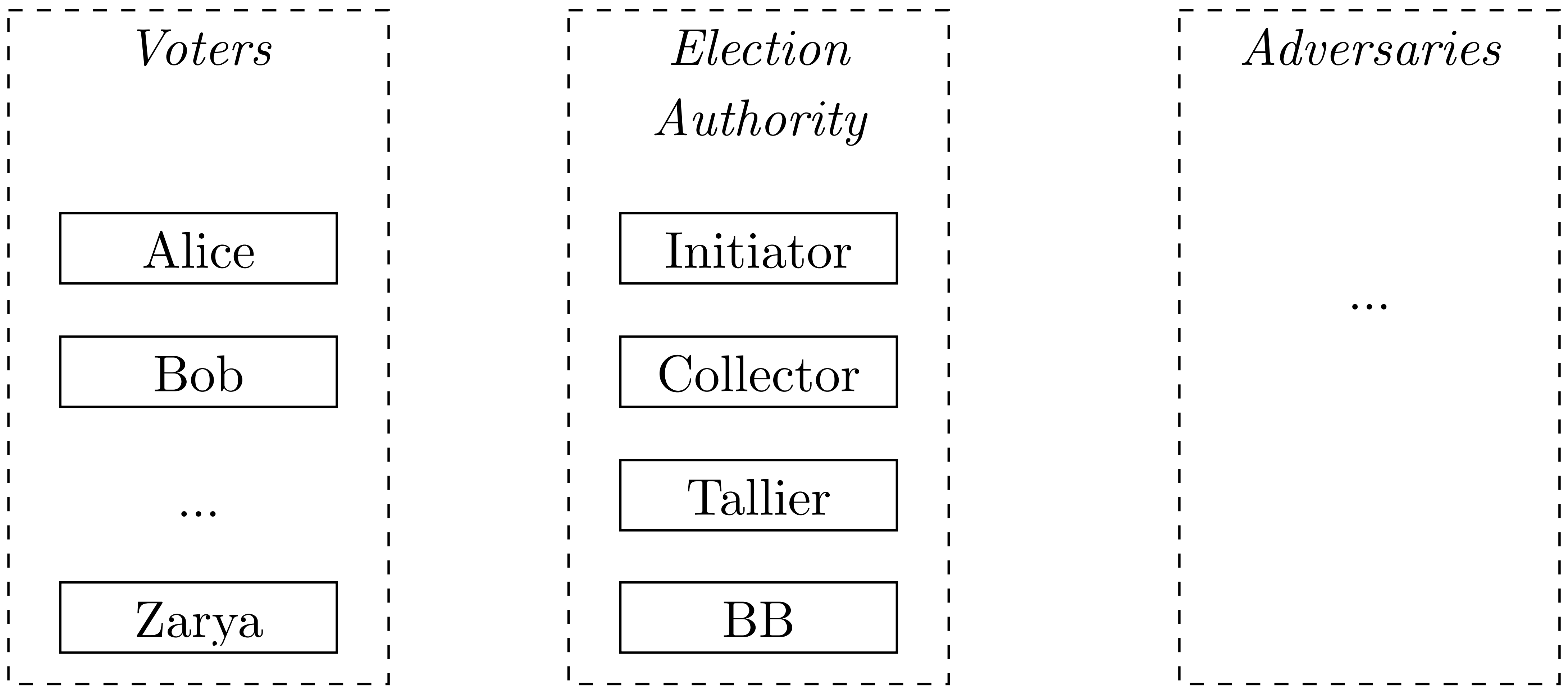
[39] Hubbers et al. [1] Ryan and Teague [40] Valimised [41] Moore and Sawhney
[34] Clarkson et al. [32] Ryan et al. [10] Becker et al. [2] Chondros et al.

Common threats to online voting systems

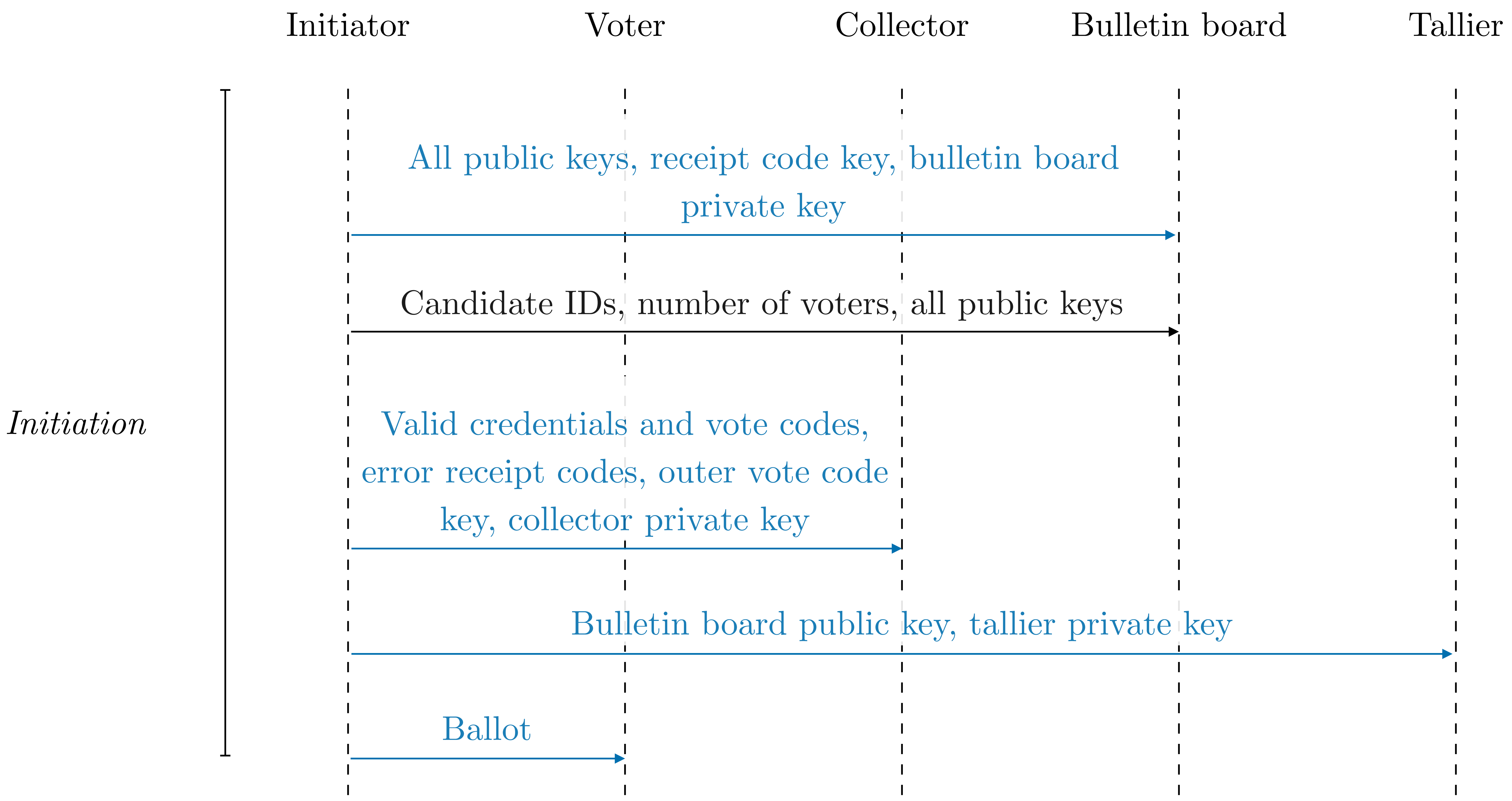
Client-side	Server-side
Over-the-shoulder coercion	Eavesdropper information disclosure
Italian attack (coercion)	MITM vote tampering
Voter spoofing	Vote disclosure
Election authority phishing	Vote and/or count tampering
Client-side tampering	Repudiation attack
Client-side information disclosure	Denial of service

Protocol Design

System agents

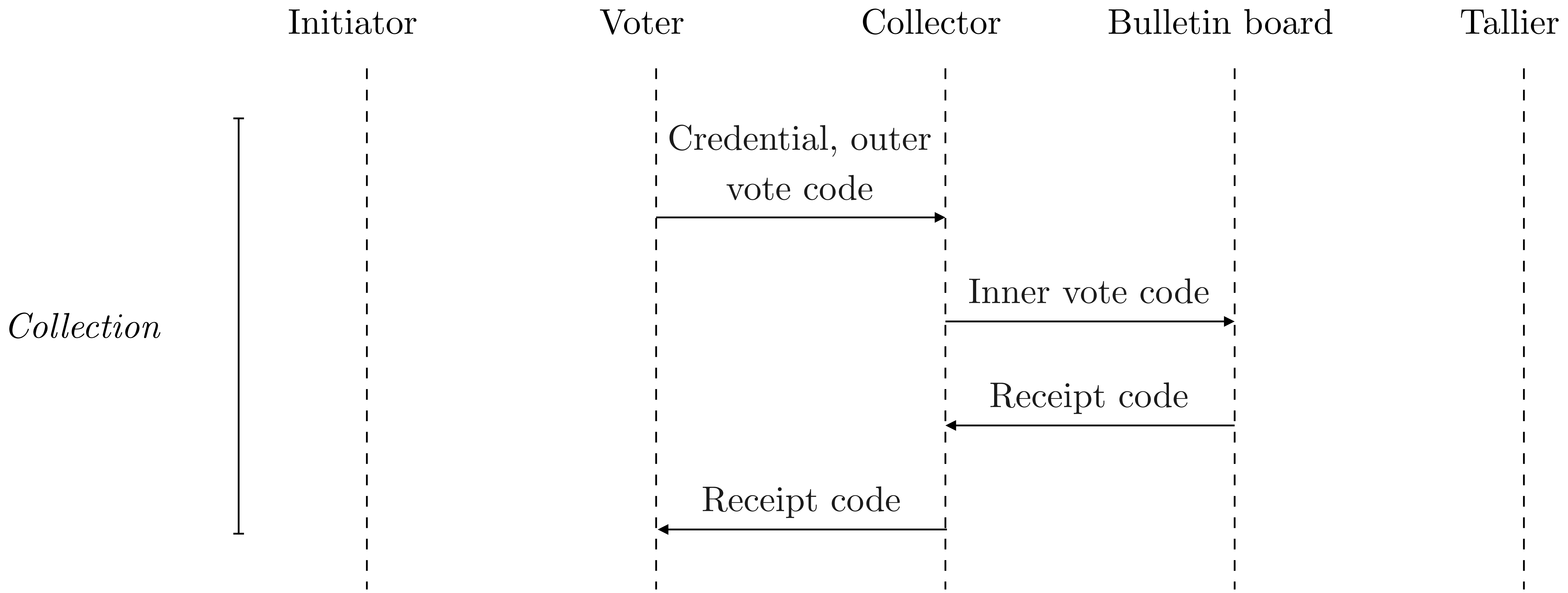


Protocol sequence (Initiation)



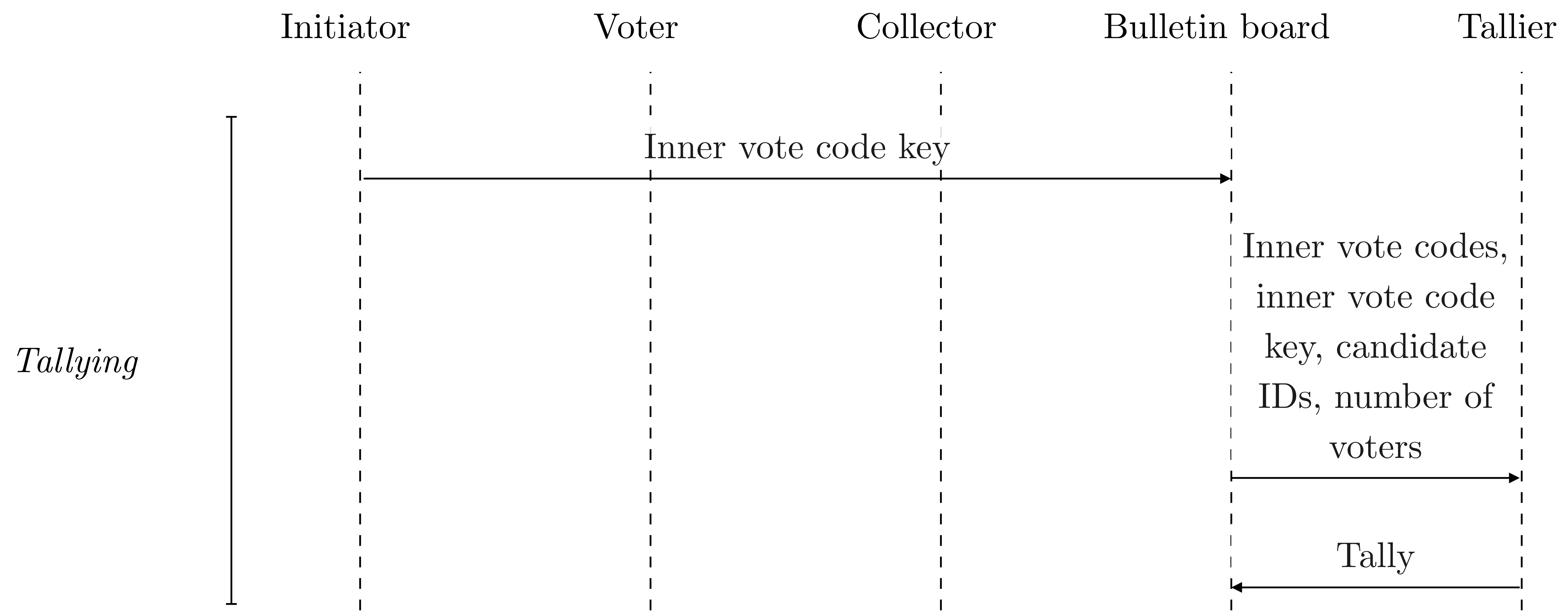
Communication coloured in blue is assumed carried out via a secure channel. All other communication can be public.

Protocol sequence (Collection)



Communication coloured in blue is assumed carried out via a secure channel. All other communication can be public.

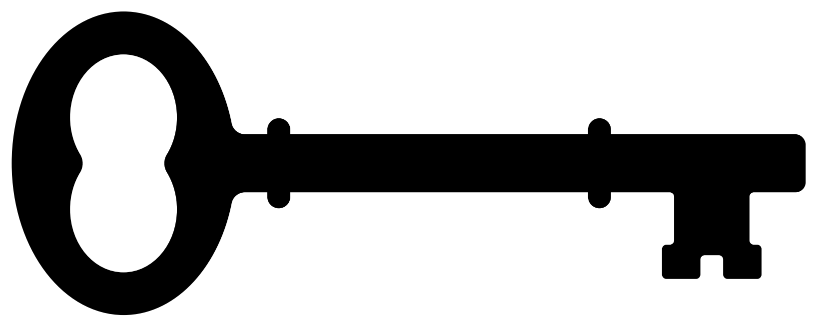
Protocol sequence (Tallying)



Communication coloured in blue is assumed carried out via a secure channel. All other communication can be public.

Threat model assumptions

- Initiator keeps private keys confidential
- Initiator generates fair ballots
- Initiator distributed a ballot to every eligible voter
- There exists a secure channel between the initiator and other election authority agents
- Collector does not disclose credential and decrypted outer vote code pairing
- There exists an append-only bulletin board
- Election authority agents do not collude
- Adversaries may perform any polynomial time computation
- Underlying secure cryptographic primitives exist
- Any coercer does not control a voter for the entire duration of an election
- There exists a separate secure channel (e.g. physical meeting) for voters to contact the election authority



Example ballot

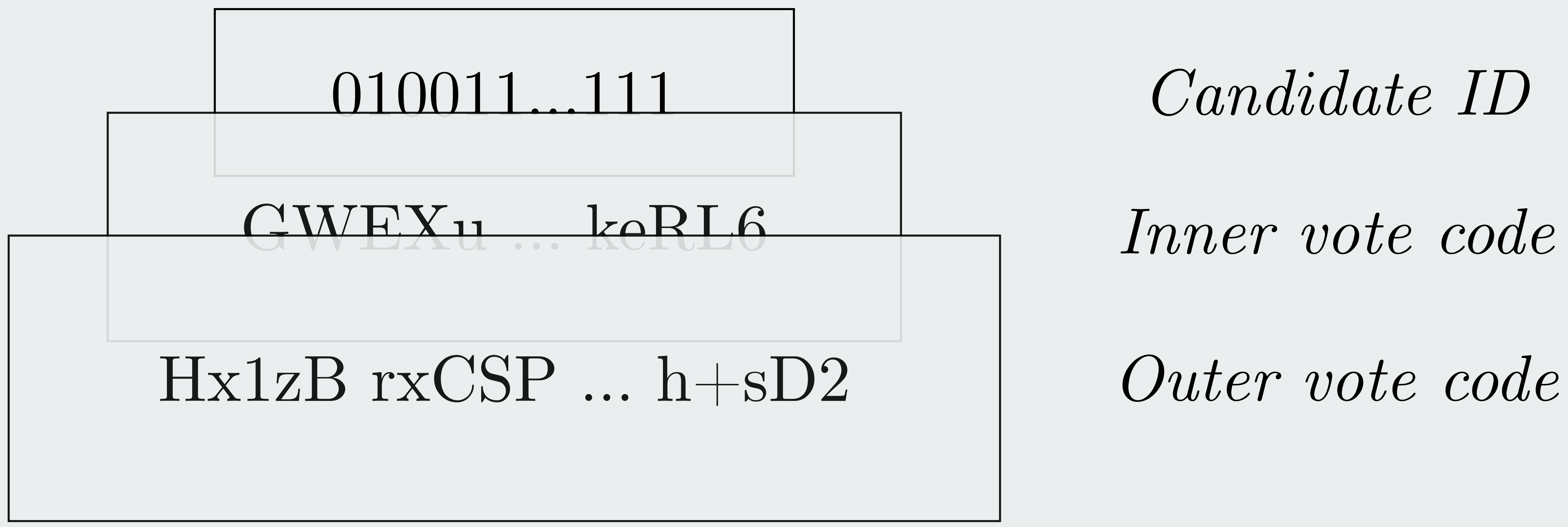
Credentials	WgDn8 hUTfw
Error receipt code	wJbhM skQIj

Candidate	*Vote code*	*Receipt code*
Anthony	e7vRA 2pCJt	YIBMj 3M0ZW
Barbara	8HszB rxCSN	jkOeb F1cGV
...	...	...
Ziggy	oBryjS WB1m	DLn/h o8fLC

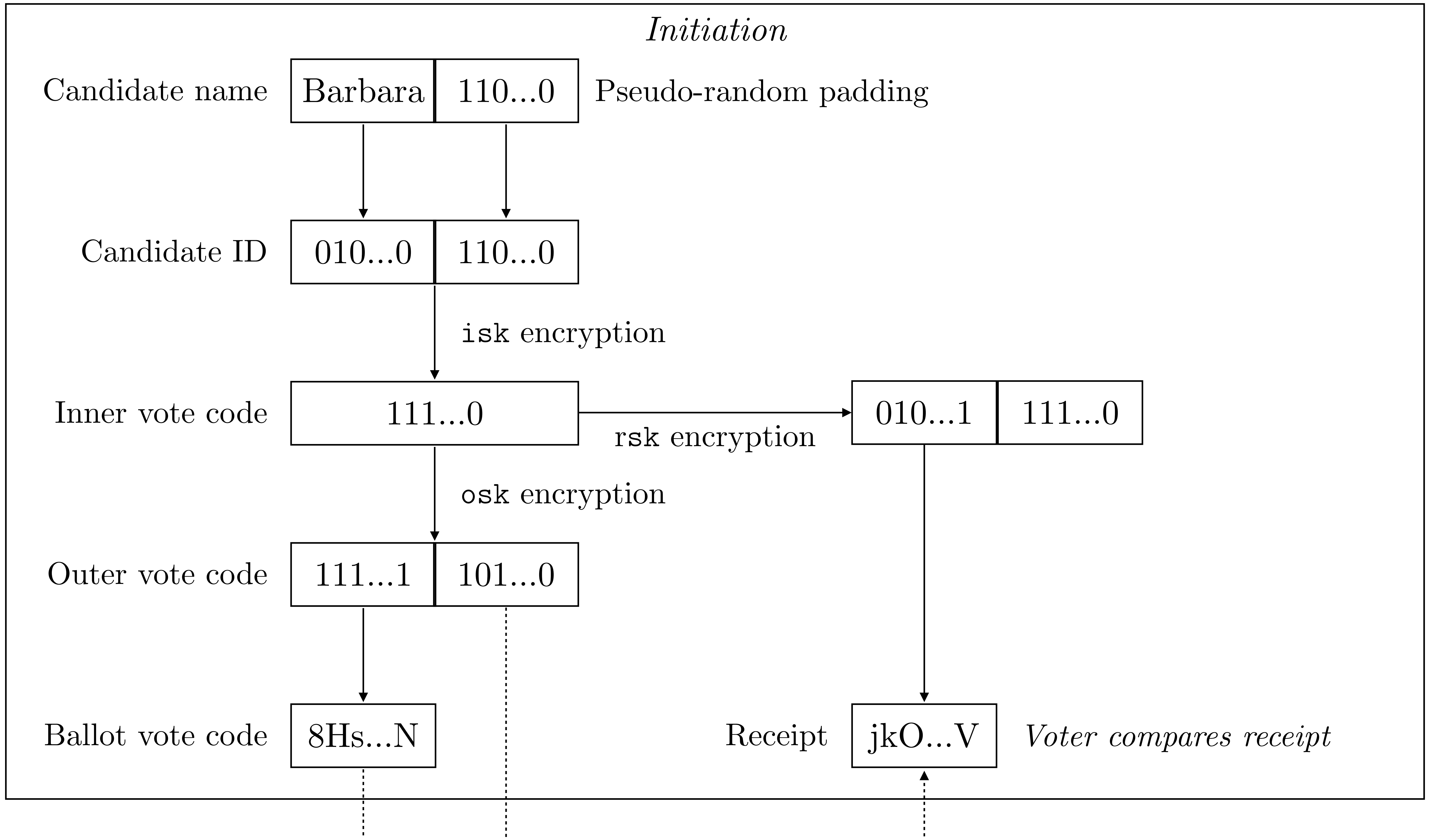
Bulletin board public key	j+IvXg+z4vdB3snqmKFSn...W

Vote code composition

- Each candidate is mapped to some binary string ID and padded with pseudo-random bits
- The padded candidate IDs are encrypted using `isk` to make the *inner vote codes*
- The inner vote codes are encrypted using `osk` to make the *outer vote codes*
- Each padded candidate ID is unique
- Following unique candidate IDs, all inner vote codes and outer vote codes are unique

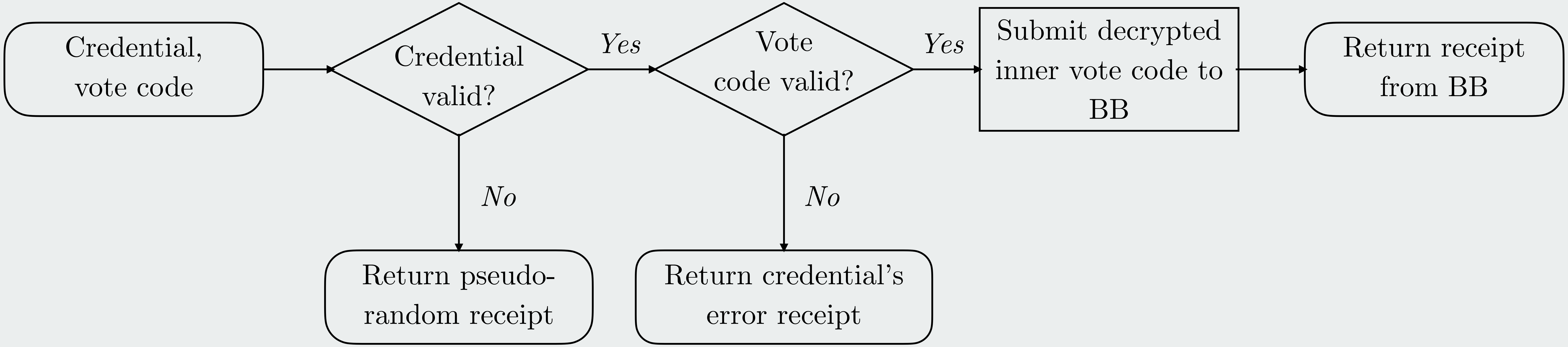


Protocol cryptography (1/3)



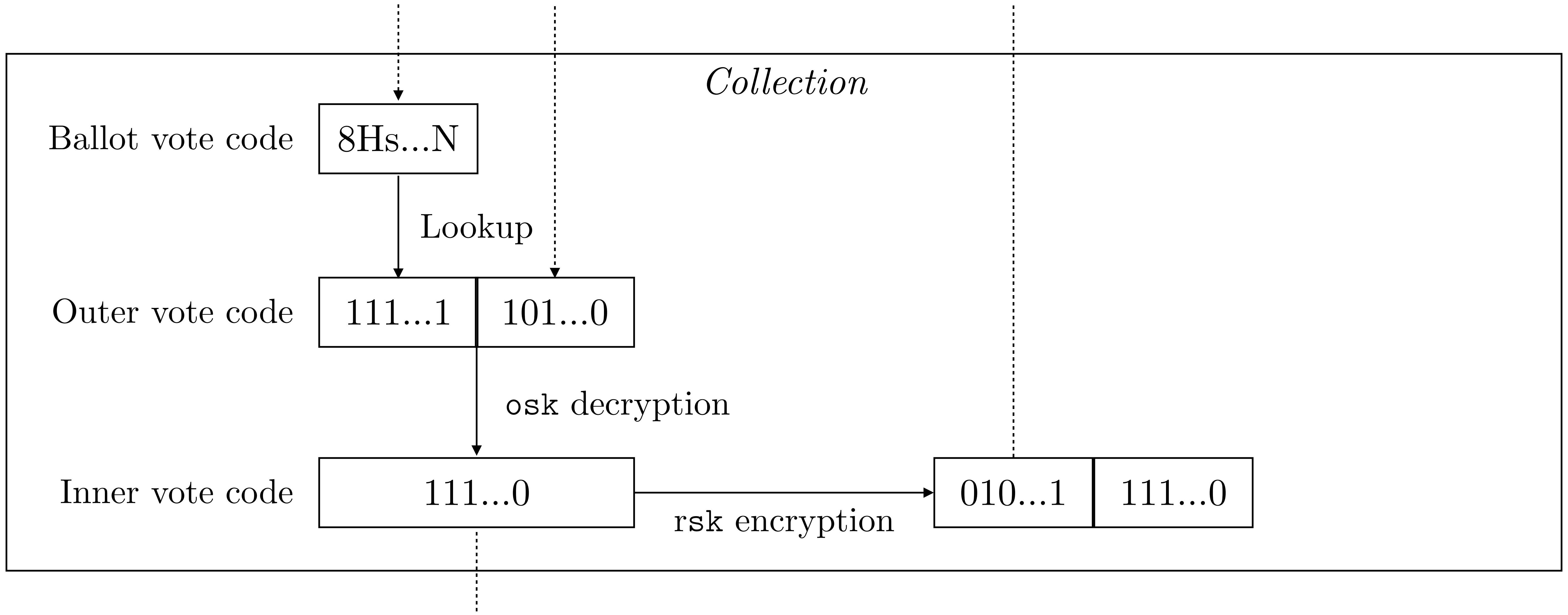
(continues on the next slide)

Collector logic flow



Protocol cryptography (2/3)

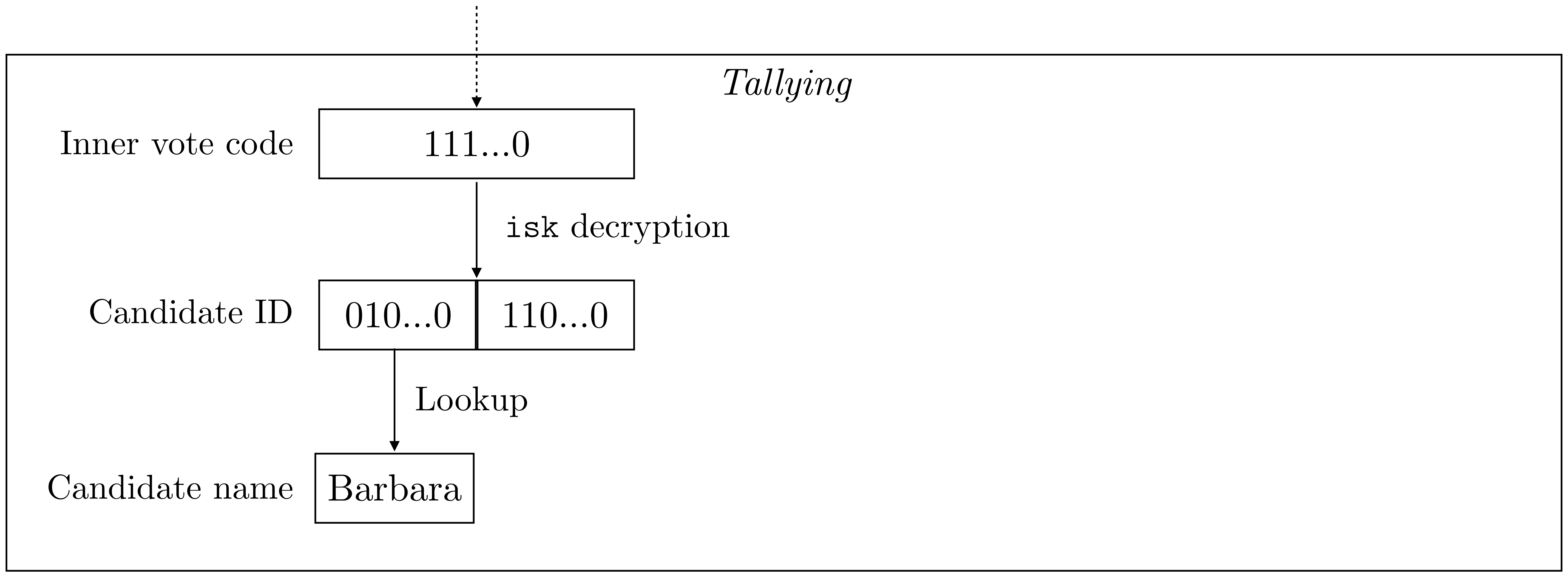
(continuing from the previous slide)



(continues on the next slide)

Protocol cryptography (3/3)

(continuing from the previous slide)



Bulletin board

- Underlying design due to Heather and Lundin [46]
- Three key properties:
  - (I) Unalterable history
  - (II) Certified publishing
  - (III) Timely publication
- Only the initiator, collector and tallier are allowed to write to the bulletin board
- Anyone can read from the bulletin board and verify its consistency

$$Entry_i = < m_i,\ t_i^W,\ w_i,\ H_i,\ S_i^W,\ t_i^B,\ S_i^B >$$

$$H_i = \mathcal{H}(m_i,\ t_i^W,\ w_i,\ H_{i\ -\ 1})$$

m_i : message,

t_i^W : writer timestamp,

w_i : writer identity,

H_i : entry hash,

S_i^W : writer signature,

t_i^B : bulletin board timestamp,

S_i^B : bulletin board signature

Security Analysis

Summary of threat analysis

- The security of the voting protocol was analysed in light of common threats to online voting systems and the threat model assumptions
- The protocol comes far in providing sufficient security for a voting system
- No reliance on client side system integrity
- Appropriately set vote code lengths make inferring or guessing votes intractable
- However, some of the assumptions made are rather strong. Most noteworthy, the assumption of complete trust in the election authority initiator may not hold in a full-scale democratic election

- Particular threats include:
  - Initiator compromise
  - Collector compromise
  - Corrupt bulletin board
  - Secret ballot breach
  - Code guessing
  - Denial of service
  - Coercion

Initiator compromise

- The most serious threat to a code voting system may be argued to be leakage of vote codes. Leaks of this form can potentially enable an eavesdropper to interpret votes, violating secret ballots, or a MITM adversary to tamper with votes
- Closely related to the vulnerability of code leakage is the similarly serious vulnerability of key leakage. Were the initiator to be compromised, the impersonation of any election authority is fair game as what gives these agents their privilege are the secret keys they hold

Collector compromise

- If the collector were to disclose associations between credentials and bulletin board published inner vote codes (or similarly the outer vote code key), receipt freedom would be at risk. A voter could could prove which inner vote code they caused to be published on the bulletin board and later decrypted to a particular candidate choice
- Should the collector attempt to tamper with the outer vote codes submitted by a voter, it will not be able to generate the receipt code expected in return by the voter. The same argument applies to dropping votes

Corrupt bulletin board

- A corrupt bulletin board inclined to collude with the collector could easily issue receipt codes in response to the inner vote code decrypted from the vote code submitted by the voter, while appending to its history a different vote code
- To reduce the dependence on the assumption that the collector and bulletin board do not collude, the bulletin board could be distributed to more closely resemble an immutable distributed ledger, or blockchain
- Such a measure would also enhance the bulletin board's resilience to denial of service attacks, as read and write requests could be services by any of the distributed entities

Secret ballot breach

- There is a vulnerability in that an adversary with a good overview of the network traffic may be able to learn the inner vote code associated with the voter's vote from the timing of a voter's vote submission, bulletin board addition and voter's receipt reception
- The current bulletin board design, due to Heather and Lundin [46], does not support delaying bulletin board entries from being written or shuffling the order of these, yet such measures could help mitigate the risk of inferring votes from vote submission and bulletin board writing times

Code guessing

- For the suggested 10 base64-encoded characters per code, there are more than a quadrillion quadrillion1 possible credential and vote code pairs
- To mitigate any chance of successfully guessing vote or receipt codes for a credential, we suggest the easily implementable mitigating action of allowing at most a certain number of vote submission attempts for any credential
- Guessing the longer cryptographic vote codes is assumed intractable, as is the forging of signatures and breaking of hashes

Denial of service

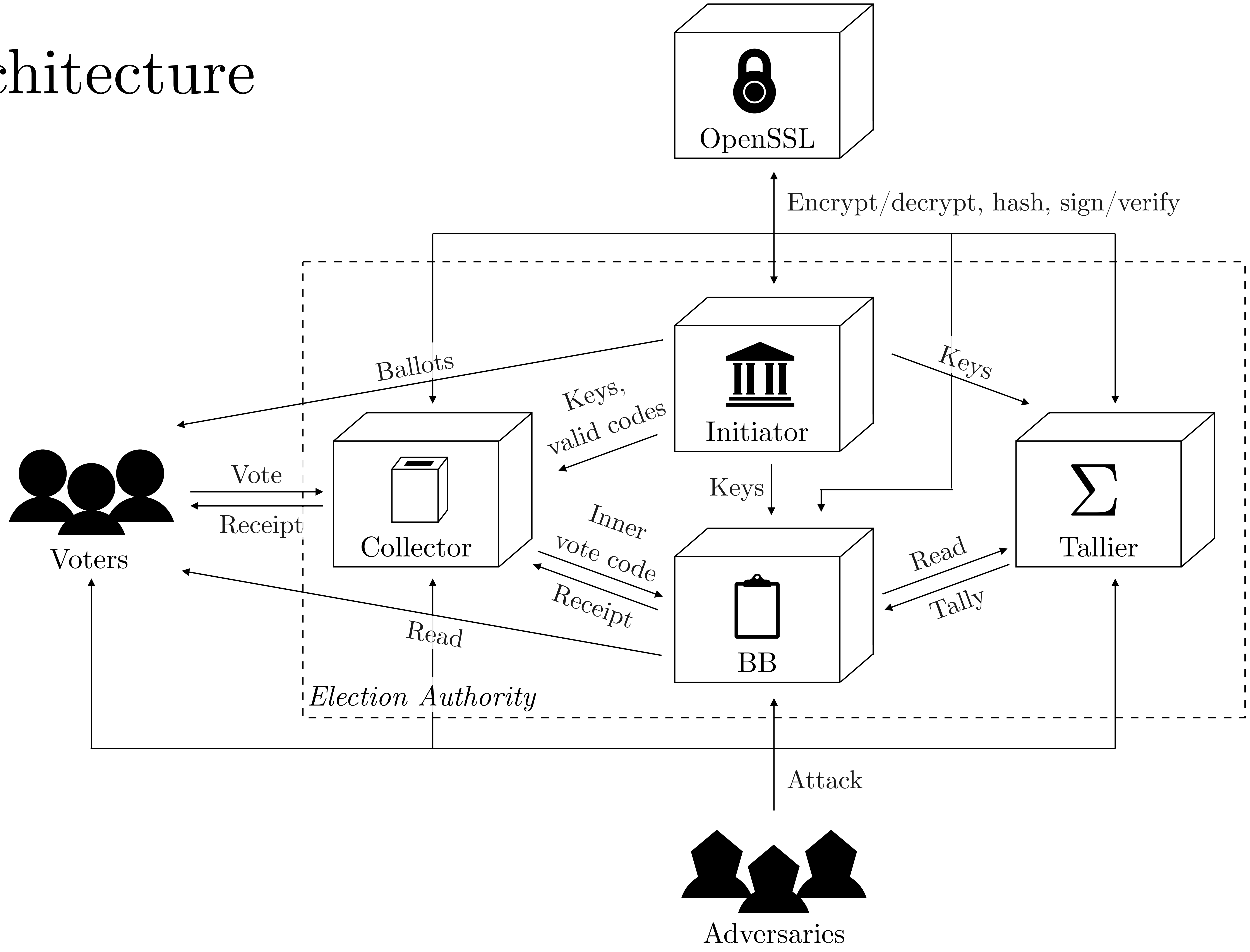
- Denial of service is an unavoidable threat to online voting systems
- We have already discussed the distribution of both collector and bulletin board, which could both help to alleviate DoS attacks
- Another threat is that of targeted DoS, for instance targeting geographic areas with a known political bias. To mitigate this, beyond the efforts of distributed collectors, we suggest a backup alternative way of voting

Coercion

- It is hard to argue that coercion may be avoided completely without a physically controlled environment, such as a pollsite, if even it can be avoided there. To the best of our knowledge, there exists no provably secure way of ensuring a remote client is not under coercion
- Rather than attempting to detect coercion directly, we suggest like many other have done before us (e.g. [34, 40]) the possibility to overwrite votes at a later stage
- In contrast to previous approaches, we suggest that such overwriting is done by contacting the election authority and casting a vote in a safe environment like a pollsite or similar.

Implementation

Software architecture



Example OpenSSL helper function

```
ByteArray hashMsg(const ByteArray& msg) {
  EVP_MD_CTX* ctx = EVP_MD_CTX_new();
  assert(EVP_DigestInit_ex(ctx, EVP_sha256(), NULL));
  assert(EVP_DigestUpdate(ctx, msg, msg.getLen()));
  ByteArray digest = ByteArray(SHA256_DIGEST_LENGTH);
  unsigned int len = digest.getLen();
  assert(EVP_DigestFinal_ex(ctx, digest, &len));
  EVP_MD_CTX_free(ctx);
  return digest;
}
```

Main classes

```
Collector
- behavior: CollectorBehaviour
- csk: ByteArray
- osk: ByteArray
- bbpk: ByteArray
- bulletinBoard: BulletinBoard*
- validVoteCodes: map<ByteArray,
                      map<ByteArray, ByteArray> >
- errorReceipCodes: map<ByteArray, ByteArray>
- spentCredentials: set<ByteArray>
+ setup(data: CollectorData): void
+ collectVote(credential: ByteArray,
              voteCode: ByteArray,
              receipt: ByteArray): void
```

```
Initiator
- isk: ByteArray
- insk: ByteArray
- bbpk: ByteArray
- bulletinBoard: BulletinBoard*
+ initiate(data: InitatorData): void
+ publishIsk(): void
```

```
BulletinBoard
- bbsk: ByteArray
- inpk: ByteArray
- cpk: ByteArray
- tpk: ByteArray
- rsk: ByteArray
- entries: vector<BulletinBoardEntry>
+ setup(data: BulletinBoardData): void
+ read(): BulletinBoardHistory
+ write(entry: BulletinBoardEntry):
                   BulletinBoardHistory
```

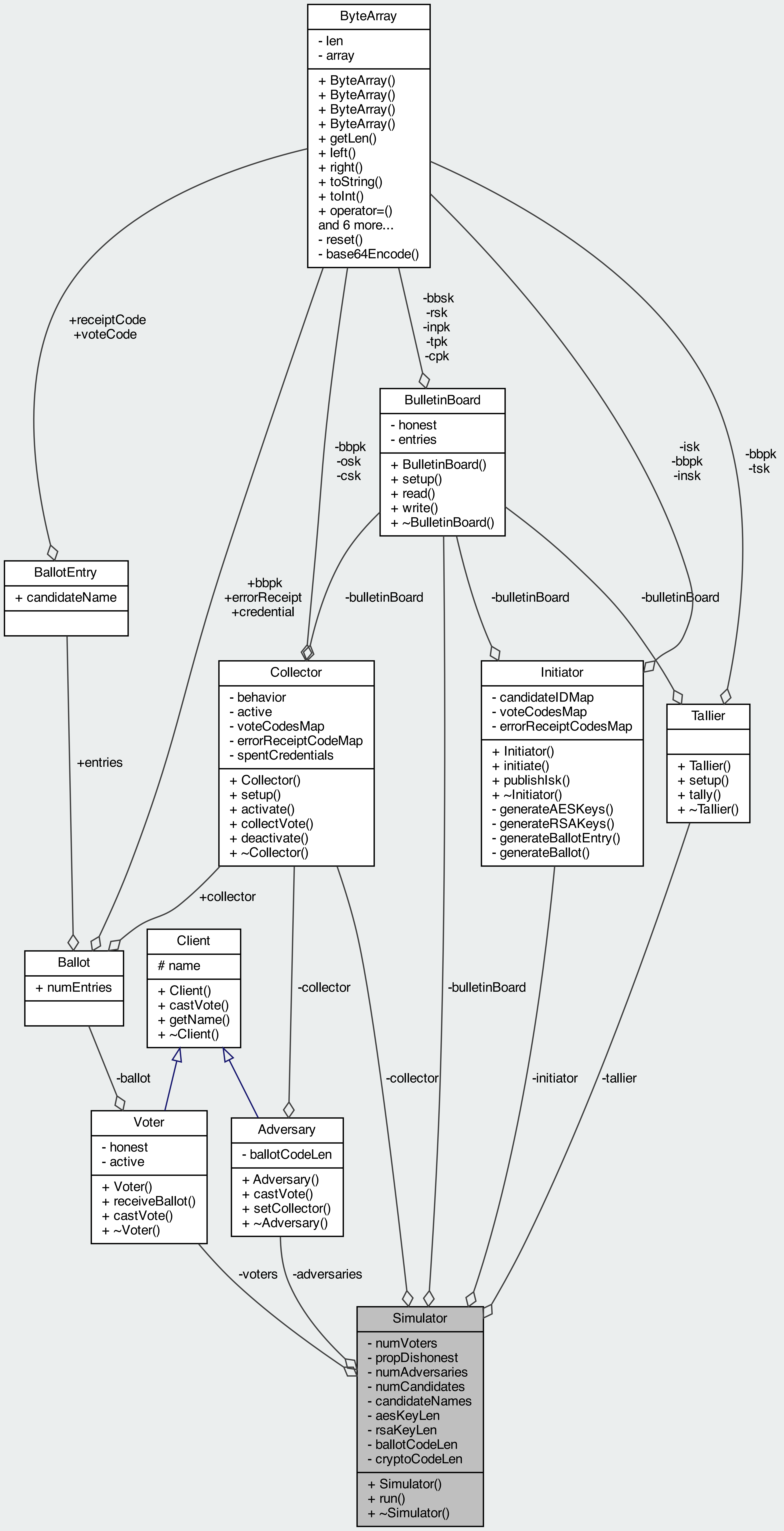
```
Voter
- honest: bool
- ballot: Ballot
+ receiveBallot(ballot: Ballot): void
+ castVote(): void
```

```
Tallier
- tsk: ByteArray
- bbpk: ByteArray
- bulletinBoard: BulletinBoard*
+ setup(data: TallierData): void
+ tally(): void
```

```
Adversary

+ castVote(): void
```

UML diagram



ByteArray
- len
- array
+ ByteArray()
+ ByteArray()
+ ByteArray()
+ ByteArray()
+ getLen()
+ left()
+ right()
+ toString()
+ toInt()
+ operator=()
and 6 more...
- reset()
- base64Encode()
+receiptCode
+voteCode
-bbsk
-rsk
-inpk
-tpk
-cpk
BulletinBoard
- honest
- entries
+ BulletinBoard()
+ setup()
+ read()
+ write()
+ ~BulletinBoard()
-bbpk
-osk
-csk
-isk
-bbpk
-insk
-bbpk
-tsk
BallotEntry
+ candidateName
+bbpk
+errorReceipt
+credential
-bulletinBoard
-bulletinBoard
-bulletinBoard
Collector
- behavior
- active
- voteCodesMap
- errorReceiptCodeMap
- spentCredentials
+ Collector()
+ setup()
+ activate()
+ collectVote()
+ deactivate()
+ ~Collector()
Initiator
- candidateIDMap
- voteCodesMap
- errorReceiptCodesMap
+ Initiator()
+ initiate()
+ publishIsk()
+ ~Initiator()
- generateAESKeys()
- generateRSAKeys()
- generateBallotEntry()
- generateBallot()
Tallier
+ Tallier()
+ setup()
+ tally()
+ ~Tallier()
+entries
+collector
Ballot
+ numEntries
Client
name
+ Client()
+ castVote()
+ getName()
+ ~Client()
-collector
-bulletinBoard
-ballot
-collector
-initiator
-tallier
Voter
- honest
- active
+ Voter()
+ receiveBallot()
+ castVote()
+ ~Voter()
Adversary
- ballotCodeLen
+ Adversary()
+ castVote()
+ setCollector()
+ ~Adversary()
-voters
-adversaries
Simulator
- numVoters
- propDishonest
- numAdversaries
- numCandidates
- candidateNames
- aesKeyLen
- rsaKeyLen
- ballotCodeLen
- cryptoCodeLen
+ Simulator()
+ run()
+ ~Simulator()

Test code coverage

LCOV - code coverage report

Current view:	top level - main		Hit	Total	Coverage
Test:	test_coverage.info	Lines:	767	806	95.2 %
Date:	2020-08-08 15:12:36	Functions:	98	101	97.0 %

Filename	Line Coverage ⇕		Functions ⇕	
adversary.cpp	100.0 %	15 / 15	100.0 %	6 / 6
ballot.cpp	97.3 %	36 / 37	100.0 %	2 / 2
bulletinBoard.cpp	91.1 %	82 / 90	100.0 %	8 / 8
bulletinBoardEntry.cpp	93.8 %	15 / 16	100.0 %	7 / 7
byteArray.cpp	99.2 %	117 / 118	93.1 %	27 / 29
client.cpp	100.0 %	6 / 6	80.0 %	4 / 5
collector.cpp	88.1 %	59 / 67	100.0 %	8 / 8
helpers.cpp	100.0 %	93 / 93	100.0 %	5 / 5
initiator.cpp	98.3 %	119 / 121	100.0 %	10 / 10
params.cpp	100.0 %	27 / 27	100.0 %	3 / 3
simulator.cpp	100.0 %	75 / 75	100.0 %	5 / 5
tallier.cpp	83.6 %	92 / 110	100.0 %	6 / 6
voter.cpp	100.0 %	31 / 31	100.0 %	7 / 7

Generated by: LCOV version 1.14

Demo

```
$ make
$ ./bin/main 10 0 0 3 16 2048 16 0 1
```

Number of voters
Proportion of dishonest voters
Number of adversaries
Number of candidates
AES key length
RSA key length
Cryptographic code length
Collector behaviour (0: honest, 1: drop, 2: tamper)
Bulletin board behaviour (0: dishonest, 1: honest)

Dishonest voter

- A dishonest voter attempts to resubmit her vote in favour of her chosen candidate
- This misbehaviour is detected and report by the collector

```
18:33:10.404 [trace] Collector activated
18:33:10.404 [trace] Voter 0 (dishonest) chose vote code: XPnyHnnECg
18:33:10.404 [trace] Collector found valid vote code, forwarded inner vote code to BB
18:33:10.410 [debug] Added BB entry: <Inner vote code, Ub7m..., Rsgy..., Collector, 9Y3o..., F4Ag..., Rsgy..., Gw4J...>
18:33:10.411 [trace] Voter 0 (dishonest) received expected receipt code: EY9HI6mStQ
18:33:10.411 [warning] Credential 3QDnEpOOEQ already voted
18:33:10.411 [warning] Voter 0 (dishonest) received inexpected receipt code: hRblAyXLrg
18:33:10.411 [trace] Voter 1 (honest) chose vote code: mwqrnwoFHA
```

Adversary

- An adversary guesses a credential and vote code and attempts to submit a vote
- Like with a dishonest voter, the collector report adversary attempts at voting

```
18:35:48.828 [warning] Collector did not find voter credential: GzTvyVJgog
18:35:48.828 [trace] Adversary 0 attempted to guess a vote and received receipt code: RDiwZCXF6A
```

Dishonest collector

- A dishonest collector may either drop (a) or attempt to tamper (b) with submitted votes
- Voters report any unexpected receipt code from the collector, and the tallier may detect vote dropping

```
18:37:23.322 [warning] Dishonest collector dropped vote from credential: hTEe19wz8g
18:37:23.322 [warning] Voter 1 (honest) received inexpected receipt code: AAAAAAAAAA
18:37:23.322 [info] Collection completed
18:37:23.322 [info] Tallying begun
18:37:23.328 [debug] Added BB entry: <isk, xWxE..., Q8ky..., Initiator, X6z8..., Rx/3..., Q8ky..., G7jU...>
18:37:23.331 [trace] Initiator published isk
18:37:23.333 [trace] Tallier read bulletin board history
18:37:23.342 [error] Tallier counted unexpected number of votes
```

(a)

```
18:39:17.658 [trace] Voter 1 (honest) chose vote code: s41GOUoDMg
18:39:17.658 [warning] Dishonest collector tampered with vote from credential: sZSH17IaKA
18:39:17.664 [debug] Added BB entry: <Inner vote code, QSrG..., tcky..., Collector, VlgS..., fTz4..., tcky..., fWBy...>
18:39:17.666 [warning] Voter 1 (honest) received inexpected receipt code: ROXaK81sXQ
```

(b)

Dishonest bulletin board

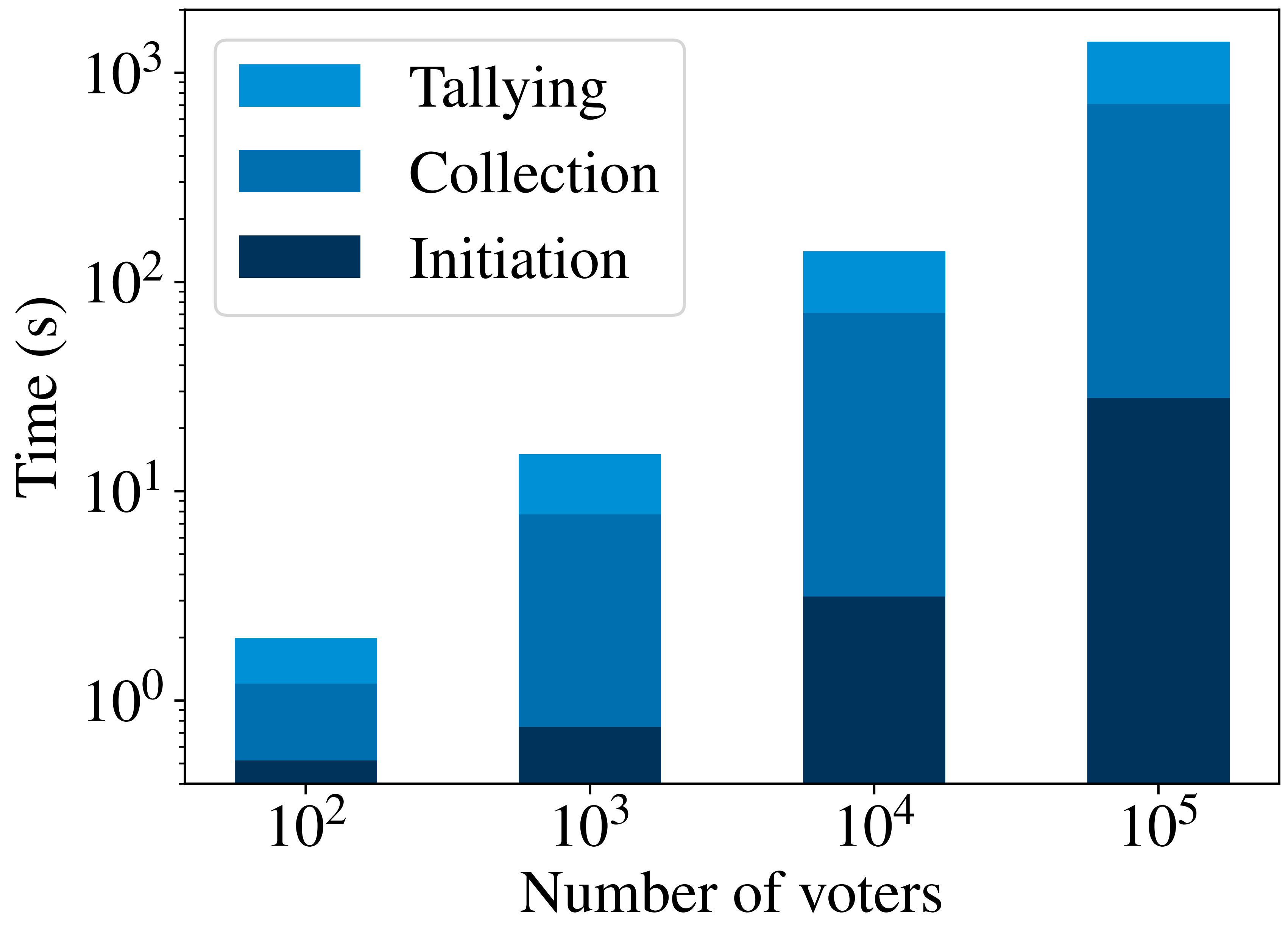
- A dishonest bulletin board attempts to modify the history of bulletin board entries
- Any reader is able to detect such modification by resulting inconsistencies in the entry timestamps, hashes and/or signatures

```
18:40:59.305 [trace] Initiator published tsk
18:40:59.315 [trace] Tallier read bulletin board history
18:40:59.316 [error] Tallier read inconsistent history (hash)
18:40:59.316 [info] Tallying completed
```

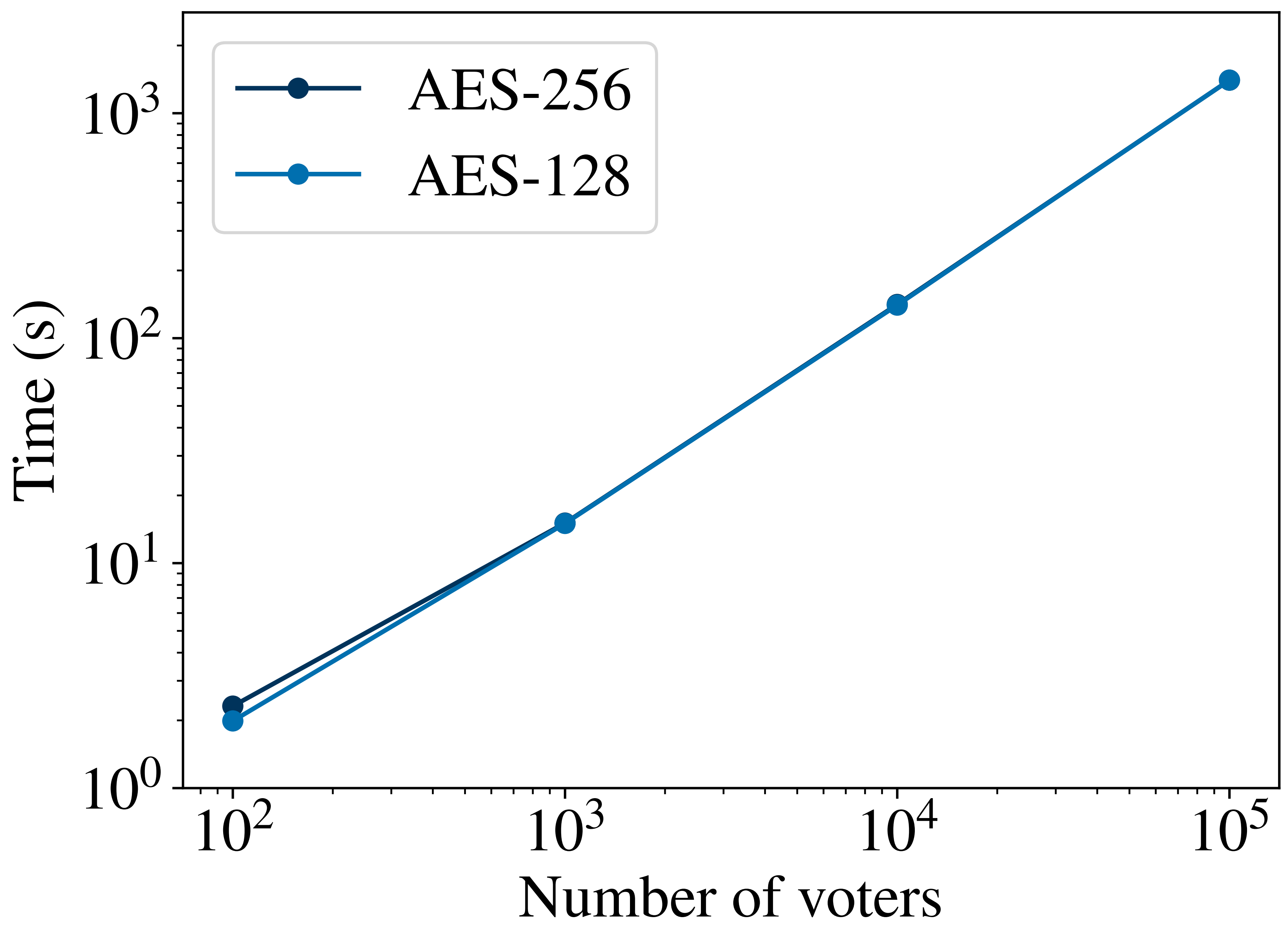
Evaluation

Distribution of time

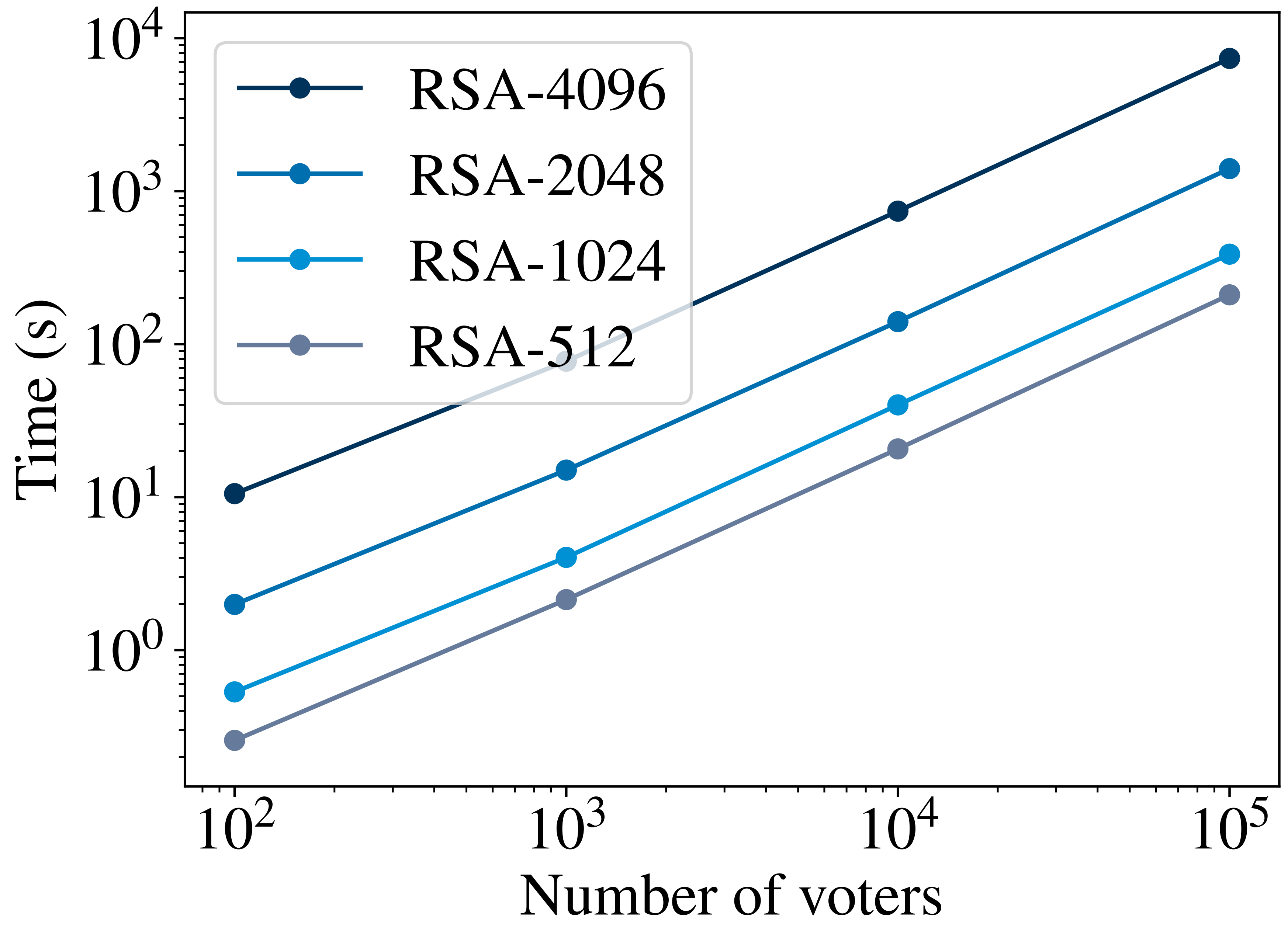
- Time increases linearly with the number of voters
- Approximately constant marginal cost of voters
- Initiation is an order of magnitude quicker than collection and tallying
- Although only 100,000 voters are tested, we argue that the linear scalability of the protocol suggests it is computationally feasible for full-scale populations of several hundred millions



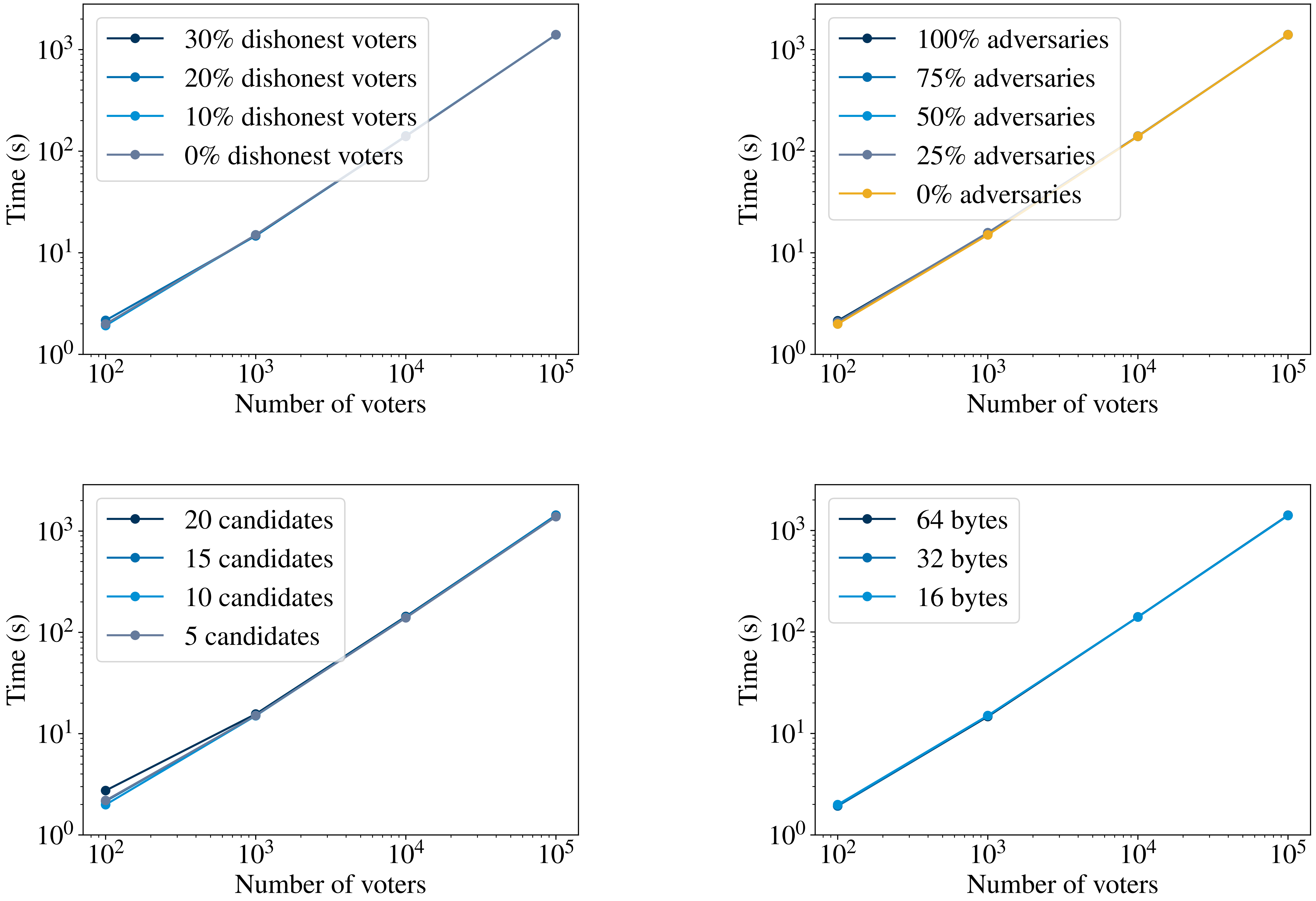
AES key length



RSA key length



Other variations



Satisfaction of requirements

- E2E verifiability under threat model assumptions, yielding software independence
- Voter may verify vote is published on the bulletin board and anyone can verify that votes published on the bulletin board are correctly tallied
- Authentication and authorisation by cryptographic credential code
- Receipt freedom assured with an honest collector
- Secret ballots with an honest initiator
- Highly usable and promising with regards to accessibility
- Only suggested coercion resistance measures, none implemented thus far
- Actions to mitigate the resilience on election authority trust have been suggested

Requirement	Satisfaction
E2E verifiability	✓
Voter authorisation and authentication	✓
Receipt freedom	✓
Secret ballot	✓
Usability and accessibility	✓
Software independence	✓
Coercion resistance	-

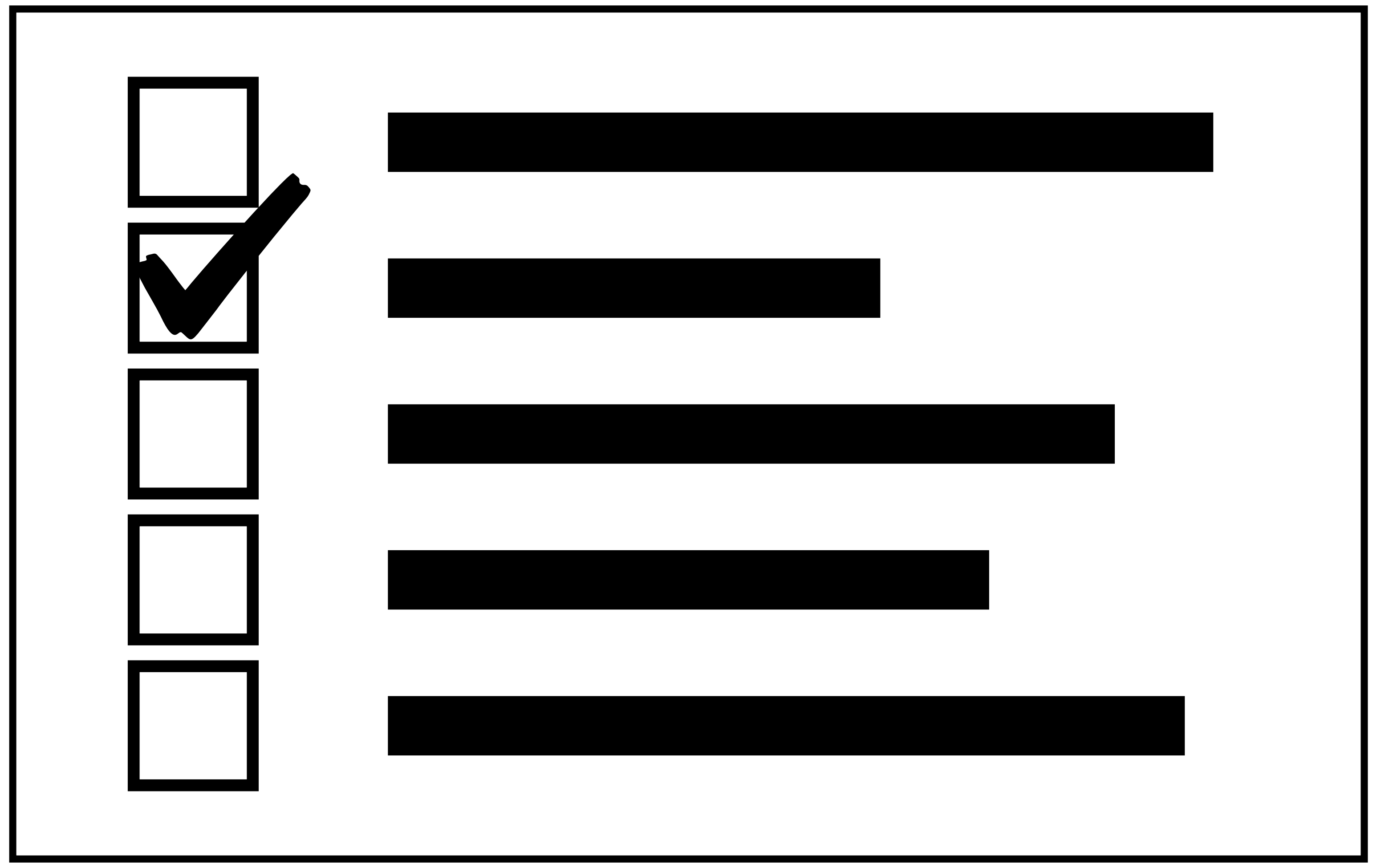
Conclusion

Achievements

- A new proof of concept, end-to-end verifiable, secure online voting protocol based on *code voting* that sets itself apart from existing systems by a new distribution of election authority trust
- The protocol is found to offer strong security against client side malware, and an intermediary has no opportunity to infer, alter or drop votes without the voter noticing
- A demonstration of the protocol gives insight about practically feasible protocol performance. A high rate of vote processing is constant across population sizes, and does not suffer considerably from strengthened symmetric encryption, nor the number of candidates or presence of adversaries
- The protocol does well with respect to all requirements for online voting systems but coercion resistance and is superior in terms of usability

Future work

- Distribution of critical key and code generation carried out by the initiator
- Threshold cryptography implementation for sensitive decryption operations (e.g. collector decryption of outer vote codes)
- Formal analysis of system security (e.g. by formal model checking)
- Survey of accessibility potential and usability improvements (e.g. inspired by online banking)
- Investigation of quantum computer secure cryptography with applications in online voting



[1] Peter YA Ryan and Vanessa Teague. Pretty good democracy. In International Workshop on Security Protocols, pages 111–130. Springer, 2009.

[2] Nikos Chondros, Bingsheng Zhang, Thomas Zacharias, Panos Diamantopoulos, Stathis Maneas, Christos Patsonakis, Alex Delis, Aggelos Kiayias, and Mema Roussopoulos. Distributed, end-to-end verifiable, and privacy-preserving inter- net voting systems. Computers & Security, 83:268–299, 2019.

[3] Bruce Schneier. American elections are too easy to hack. We must take action now. The Guardian [Internet]. 2018 Apr 18. Available from: https://g.co/ kgs/mwCgZJ [cited 2020 Aug 13].

[4] Alexa Corse. Democrats' Iowa Caucus Voting App Stirs Security Concerns. The Wall Street Journal [Internet]. 2020 Jan 26. Avail- able from: https://www.wsj.com/articles/dems-iowa-caucus-voting- app-stirs-security-concerns-11580063221 [cited 2020 Aug 13].

[5] Lily Hay Newman. The Iowa Caucus Tech Meltdown Is a Warn- ing. The Wall Street Journal [Internet]. 2020 Feb 4. Available from: https://www.wired.com/story/iowa-democratic-caucus-app-tech- meltdown-warning/ [cited 2020 Aug 13].

[6] Abby Abazorius. MIT researchers identify security vulnerabilities in voting app. MIT News [Internet]. 2020 February 13. Available from: http://news.mit. edu/2020/voting-voatz-app-hack-issues-0213 [cited 2020 Mar 30].

[7] Robert Alan Dahl. Democracy and its Critics, pages 1–9. Yale University Press, 1989.

[8] Ronnie Dugger. Annals of democracy: Counting votes. New Yorker, 64(38): 40–108, 1988.

[9] Syed Taha Ali and Judy Murray. An overview of end-to-end verifiable voting systems. In Feng Hao and Peter YA Ryan, editors, Real-World Electronic Voting: Design, Analysis and Deployment, chapter 7, pages 175–218. CRC Press, Florida, 2016.

[10] Matt Becker, Lauren Chandler, Patrick Hayes, Wesley Hedrick, Kurtis Jensen, Srini Kandikattu, Pete Martin, Scott Meier, Leopoldo Pen˜a, Kun Peng, Aleck Silva-Pinto, Jeffrey Stern, Liv Stromme, Lakshman Tavag, Dave Wallick, and Noah Zweben. Proof of vote - an end-to-end verifiable digital voting proto- col using distributed ledger technology (blockchain). Technical report, Votem Corp., Oct 2018. Available from: https://github.com/votem/proof-of-vote [cited 2020 May 27].

[11] Susan Dzieduszycka-Suinat, Judy Murray, Joseph R. Kiniry, Daniel M. Zimmer- man, Daniel Wagner, Philip Robinson, Adam Foltzer, and Shpatar Morina. The Future of Voting. End-to-end verifiable internet voting. Specification and fea- sibility assessment study. Technical report, U.S. Vote Foundation, Jul 2015. Available from: https://www.usvotefoundation.org/E2E-VIV/ [cited 2020 May 21].

[12] Matthew Bernhard, Josh Benaloh, J Alex Halderman, Ronald L Rivest, Peter YA Ryan, Philip B Stark, Vanessa Teague, Poorvi L Vora, and Dan S Wallach. Public evidence from secret ballots. In International Joint Conference on Electronic Voting, pages 84–109. Springer, 2017.

[13] David Chaum. Secret-ballot receipts: True voter-verifiable elections. IEEE se- curity & privacy, 2(1):38–47, 2004.

[14] Peter YA Ryan, David Bismark, James Heather, Steve Schneider, and Zhe Xia. Prˆet a` voter: a voter-verifiable voting system. IEEE transactions on information forensics and security, 4(4):662–673, 2009.

[15] Kevin Fisher, Richard Carback, and Alan T Sherman. Punchscan: Introduction and system definition of a high-integrity election system. In Proceedings of Workshop on Trustworthy Elections, pages 19–29, 2006.

[16] David Chaum, Aleks Essex, Richard Carback, Jeremy Clark, Stefan Popove- niuc, Alan Sherman, and Poorvi Vora. Scantegrity: End-to-end voter-verifiable optical-scan voting. IEEE Security & Privacy, 6(3):40–46, 2008.

[17] Ben Adida and Ronald L Rivest. Scratch & vote: self-contained paper-based cryptographic voting. In Proceedings of the 5th ACM workshop on Privacy in electronic society, pages 29–40, 2006.

[18] Jens-Matthias Bohli, Jo¨rn Mu¨ller-Quade, and Stefan Ro¨hrich. Bingo voting: Secure and coercion-free voting using a trusted random number generator. In International Conference on E-Voting and Identity, pages 111–124. Springer, 2007.

[19] Daniel Sandler, Kyle Derr, and Dan S Wallach. Votebox: A tamper-evident, verifiable electronic voting system. In USENIX Security Symposium, volume 4, page 87, 2008.

[20] Jonathan Ben-Nun, Niko Fahri, Morgan Llewellyn, Ben Riva, Alon Rosen, Am- non Ta-Shma, and Douglas Wikstro¨m. A new implementation of a dual (paper and cryptographic) voting system. In 5th International Conference on Electronic Voting 2012 (EVOTE2012). Gesellschaft fu¨r Informatik eV, 2012.

[21] Susan Bell, Josh Benaloh, Michael D Byrne, Dana DeBeauvoir, Bryce Eakin, Philip Kortum, Neal McBurnett, Olivier Pereira, Philip B Stark, Dan S Wallach, et al. Star-vote: A secure, transparent, auditable, and reliable voting system. In 2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13), 2013.

[22] Feng Hao, Matthew N Kreeger, Brian Randell, Dylan Clarke, Siamak F Sha- handashti, and Peter Hyun-Jeen Lee. Every vote counts: Ensuring integrity in large-scale electronic voting. In 2014 Electronic Voting Technology Workshop/- Workshop on Trustworthy Elections (EVT/WOTE 14), 2014.

[23] Computer Security Resource Center. Glossary. National Institute of Standards and Technology [Internet], 2020. Available from: https://csrc.nist.gov/ Glossary [cited 2020 Jun 4].

[24] Salil P Vadhan. Pseudorandomness. In Foundations and Trends in Theoretical Computer Science, volume 7, chapter 1, pages 2–9. now publishers, Dec 2012.

[25] Alfred J Menezes, Jonathan Katz, Paul C Van Oorschot, and Scott A Vanstone. Handbook of Applied Cryptography, chapter 1. CRC press, 1996.

[26] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2):120–126, 1978.

[27] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Tech- nical report, bitcoin.org, Oct 2008. Available from: https://bitcoin.org/ bitcoin.pdf [cited 2020 Jun 10].

[28] Adi Shamir. How to share a secret. Communications of the ACM, 22(11):612– 613, 1979.

[29] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. European transactions on Telecommunications, 8(5):481–490, 1997.

[30] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 24(2):84–90, 1981.

[31] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the estonian internet voting system. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pages 703–715, 2014.

[32] Peter YA Ryan, Peter B Rønne, and Vincenzo Iovino. Selene: Voting with trans- parent verifiability and coercion-mitigation. In International Conference on Fi- nancial Cryptography and Data Security, pages 176–192. Springer, 2016.

[33] Michael A Specter, James Koppel, and Daniel Weitzner. The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz, the First Internet Voting Application Used in US Federal Elections. MIT [Preprint]. MIT; 2020 Feb. Avail- able from: https://internetpolicy.mit.edu/wp-content/uploads/2020/ 02/ SecurityAnalysisOfVoatz_Public.pdf [cited 2020 Mar 30].

[34] Michael R Clarkson, Stephen Chong, and Andrew C Myers. Civitas: Toward a secure voting system. In 2008 IEEE Symposium on Security and Privacy (sp 2008), pages 354–368. IEEE, 2008.

[35] Mahender Kumar, Satish Chand, and CP Katti. A secure end-to-end verifiable internet-voting system using identity-based blind signature. IEEE Systems Jour- nal, 2020.

[36] Robert Kofler, Robert Krimmer, and Alexander Prosser. Electronic voting: algo- rithmic and implementation issues. In 36th Annual Hawaii International Con- ference on System Sciences, 2003. Proceedings of the, pages 7–pp. IEEE, 2003.

[37] Ronald L Rivest. On the notion of 'software independence'in voting systems. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881):3759–3767, 2008.

[38] Matthew Rosenberg, Nick Corasaniti, Sheera Frenkel, and Nicole Perlroth. Faulty Iowa App Was Part of Push to Restore Democrats' Digital Edge. The New York Times [Internet]. 2020 Feb 4. Available from: https://nyti.ms/ 2uaeZRh [cited 2020 Mar 27].

[39] Engelbert Hubbers, Bart Jacobs, and Wolter Pieters. RIES - Internet Voting in Action. In 29th Annual International Computer Software and Applications Conference (COMPSAC'05), volume 1, pages 417–424. IEEE, 2005.

[40] Valimised. General Framework of Electronic Voting and Implementation thereof at National Elections in Estonia. Technical report, State Electoral Office of Estonia, Tallinn, Jun 2017. Available from: https:// www.valimised.ee/ en/internet-voting/documents-about-internet-voting [cited 2020 Jun 4].

[41] Larry Moore and Nimit Sawhney. UNDER THE HOOD - The West Virginia Mobile Voting Pilot. Technical report, Voatz, Inc., Feb 2019.

[42] Adam Shostack. Threat modeling: Designing for security, chapter 3. John Wiley & Sons, 2014.

[43] Michael A Specter and Alex J Halderman. Security Analysis of the Democ- racy Live Online Voting System. Technical report, MIT, Jun 2020. Avail- able from: https://internetpolicy.mit.edu/wp-content/uploads/2020/ 06/ OmniBallot.pdf [cited 2020 Jun 11].

[44] IBM Quantum Experience. Shor's algorithm. IBM [Internet], 2020. Available from: https://quantum-computing.ibm.com/docs/guide/q-algos/shor-s- algorithm [cited 2020 Jun 4].

[45] IBM Quantum Experience. Grover's algorithm. IBM [Internet], 2020. Available from: https://quantum-computing.ibm.com/docs/guide/q-algos/grover- s-algorithm [cited 2020 Jun 4].

[46] James Heather and David Lundin. The append-only web bulletin board. In International Workshop on Formal Aspects in Security and Trust, pages 242–256. Springer, 2008.

[47] Chris Culnane and Steve Schneider. A peered bulletin board for robust use in verifiable voting systems. In 2014 IEEE 27th Computer Security Foundations Symposium, pages 169–183. IEEE, 2014.

[48] Rosario Gennaro, Stanisław Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. In Interna- tional Conference on the Theory and Applications of Cryptographic Techniques, pages 295–310. Springer, 1999.

[49] OpenSSL Software Foundation. OpenSSL. Version 1.1.1g. Available from: https://www.openssl.org/ [cited 2020 May 12].

[50] Elaine Barker. Recommendations for Key Management: Part 1 - General. Tech- nical report, National Institute of Standards and Technology, May 2020. Avail- able from: https://doi.org/10.6028/NIST.SP.800-57pt1r5 [cited 2020 Aug 11].