Belenios with cast-as-intended

*Véronique Cortier, **Alexandre Debant**, Pierrick Gaudry, Stéphane Glondu*

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

8th Voting workshop
Bol, Brač, Croatia

Belenios

Belenios is a protocol firstly introduced in 2014

- it extends Helios [Adida - 2008]
- it has many extensions: BeleniosVS, BeleniosRF, …
- often studied in the literature

Belenios is a software

- open source (GNU AGPLv3)
- developed in OCaml and Javascript

Belenios is a platform - https://vote.belenios.org/admin

- mainly associations and professional elections
- + 2000 elections
- + 100 000 voters

Belenios security

	Voter	Voting client	Voting server	Registrar	Trustees
Verifiability	😇	😇	👺	😇	👺
	😇	😇	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Election Verifiability with ProVerif. Cortier, Debant, and Cheval - CSF 2023
Provably Improving Election Verifiability in Belenios. Baloglu, Bursuc, Mauw, and Pang - E-Vote-ID 2021.
Belenios: A Simple Private and Verifiable Electronic Voting System. Cortier, Gaudry, and Glondu - 2019.

Belenios security

	Voter	Voting client	Voting server	Registrar	Trustees
Verifiability	😇	😇	👺	😇	👺
	😇	😇	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Election Verifiability with ProVerif. Cortier, Debant, and Cheval - CSF 2023
Provably Improving Election Verifiability in Belenios. Baloglu, Bursuc, Mauw, and Pang - E-Vote-ID 2021.
Belenios: A Simple Private and Verifiable Electronic Voting System. Cortier, Gaudry, and Glondu - 2019.

Belenios security

	Voter	Voting client	Voting server	Registrar	Trustees
Verifiability	😇	😇	👺	😇	👺
	😇	😇	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Can we do better?

Election Verifiability with ProVerif. Cortier, Debant, and Cheval - CSF 2023
Provably Improving Election Verifiability in Belenios. Baloglu, Bursuc, Mauw, and Pang - E-Vote-ID 2021.
Belenios: A Simple Private and Verifiable Electronic Voting System. Cortier, Gaudry, and Glondu - 2019.

Existing cast-as-intended solutions

- **Cast-or-audit:** e.g. Benaloh challenge [Benaloh - 2006]
- **Second device:** e.g. Estonian IVXV protocol
- **Return codes:** e.g. Swiss Post protocol
- **Transparent voting:** e.g. sElect and Selene protocols [Ryan *et al* - 2015], [Küsters *et al* - 2016]
- **Cast-and-audit:** [Cortier *et al* - 2019]

Existing cast-as-intended solutions

- **Cast-or-audit:** e.g. Benaloh challenge [Benaloh - 2006]
- **Second device:** e.g. Estonian IVXV protocol
- **Return codes:** e.g. Swiss Post protocol
- **Transparent voting:** e.g. sElect and Selene protocols [Ryan *et al* - 2015], [Küsters *et al* - 2016]
- **Cast-and-audit:** [Cortier *et al* - 2019]

Cast-and-audit

Idea of cast-and-audit

- Alice wants to vote for $X \in \{1, \dots, p\}$
- Alice picks $A \in \{1, \dots, p\}$ at random
- Alice casts ballot $b = (\{X\}_{pk}^{r_X}, \{A\}_{pk}^{r_A}, \{B\}_{pk}^{r_B}, \texttt{zkp}(B = X + A \mod n))$
- Alice randomly chooses to audit A or B

Cast-and-audit

Idea of cast-and-audit

- Alice wants to vote for $X \in \{1, \ldots, p\}$
- Alice picks $A \in \{1, \ldots, p\}$ at random
- Alice casts ballot $b = (\{X\}_{pk}^{r_X}, \{A\}_{pk}^{r_A}, \{B\}_{pk}^{r_B}, \mathtt{zkp}(B = X + A \mod n))$
- Alice randomly chooses to audit A or B

If the attacker casts a ballot $b' = (\{X'\}_{pk}^{r'_X}, c_A, c_B, \pi)$ with $X \neq X'$ then either:

- the proof π is invalid; or
- c_A does not encrypt A; or
- c_B does not encrypt B

Cast-and-audit

Idea of cast-and-audit

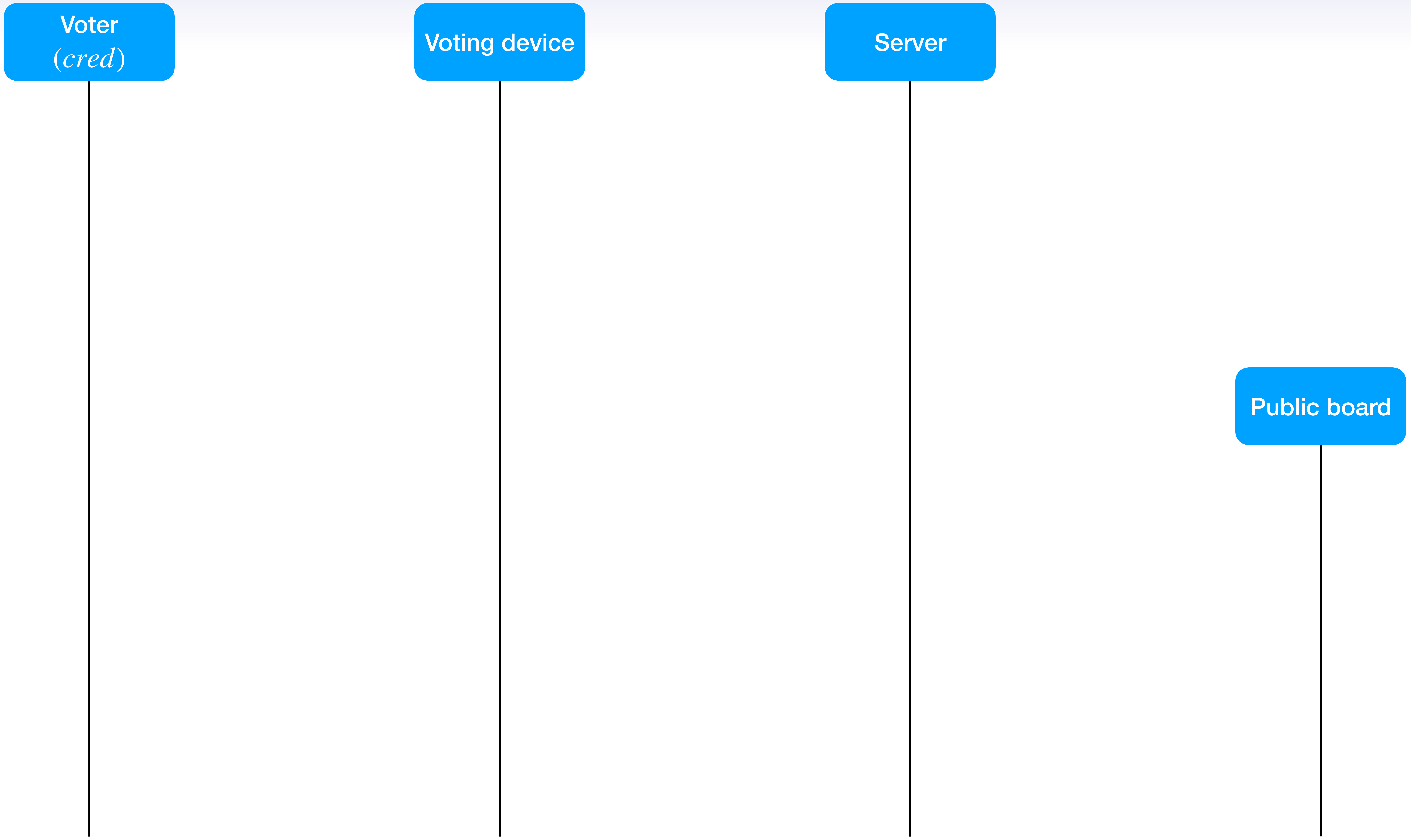
- Alice wants to vote for $X \in \{1, \dots, p\}$
- Alice picks $A \in \{1, \dots, p\}$ at random
- Alice casts ballot $b = (\{X\}_{pk}^{r_X}, \{A\}_{pk}^{r_A}, \{B\}_{pk}^{r_B}, \mathtt{zkp}(B = X + A \mod n))$
- Alice randomly chooses to audit A or B

If the attacker casts a ballot $b' = (\{X'\}_{pk}^{r'_X}, c_A, c_B, \pi)$ with $X \neq X'$ then either:

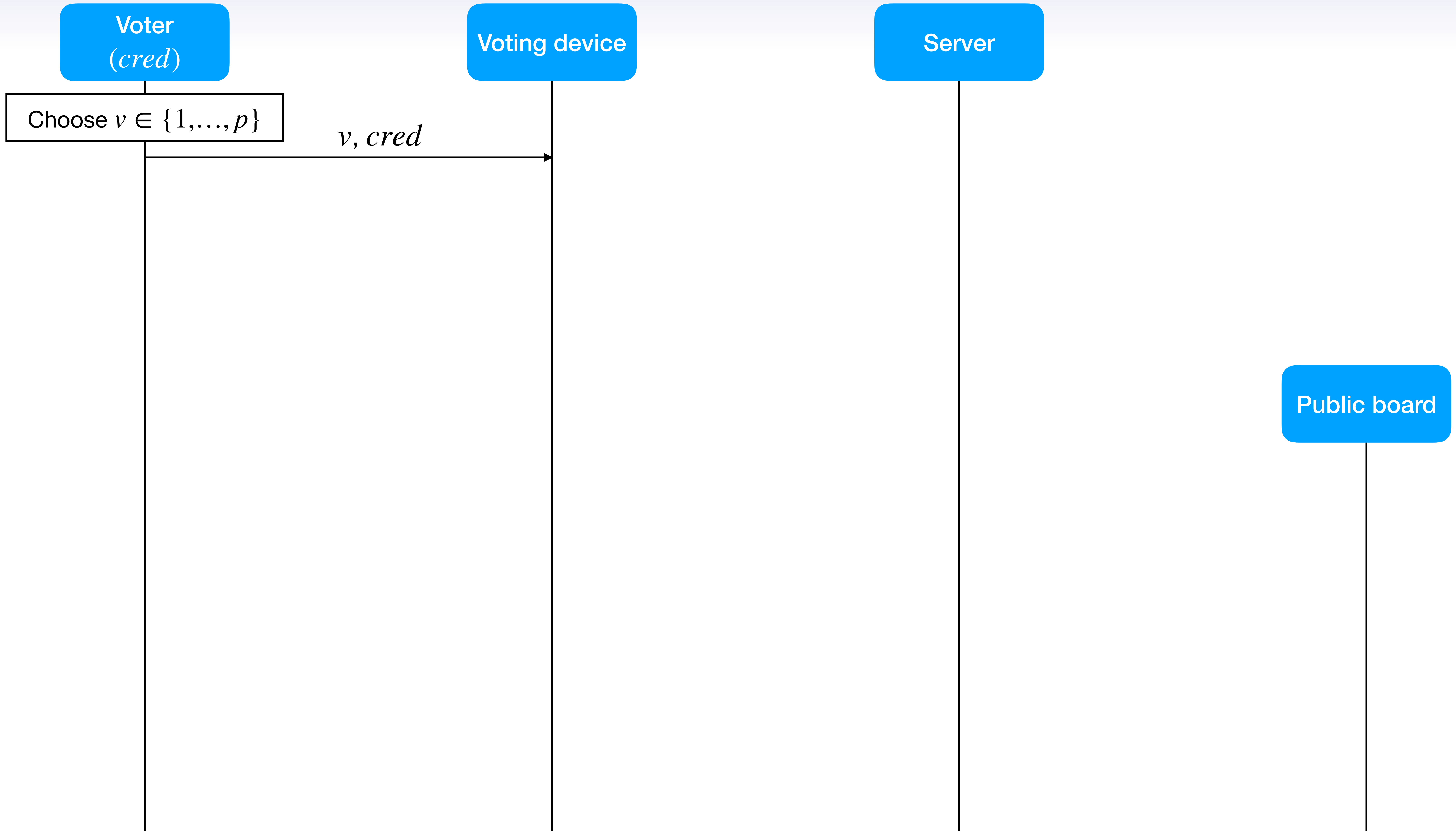
- the proof π is invalid; or
- c_A does not encrypt A; or
- c_B does not encrypt B

Alice will detect any modification of X with probability $1/2$

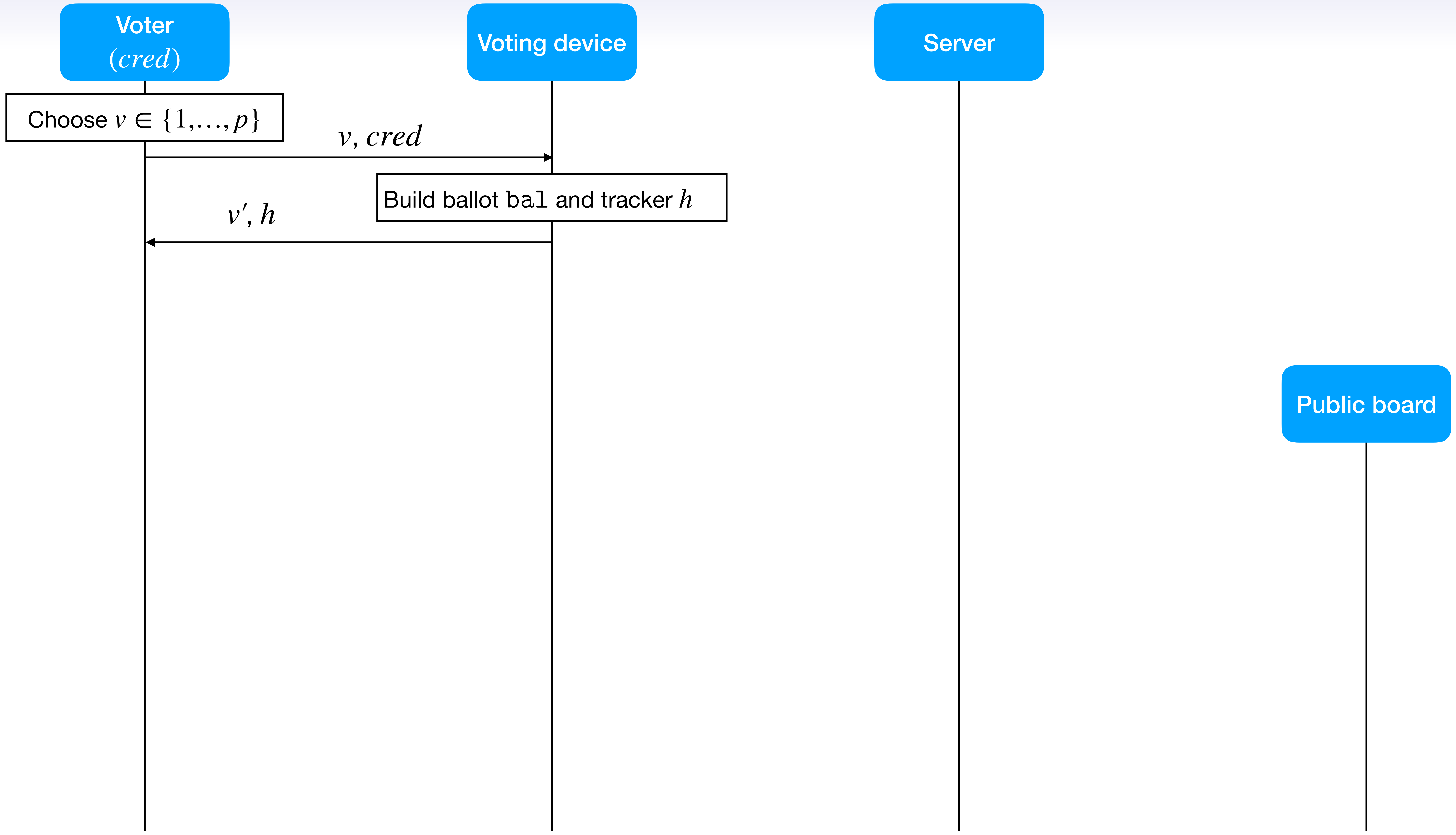
Belenios protocol



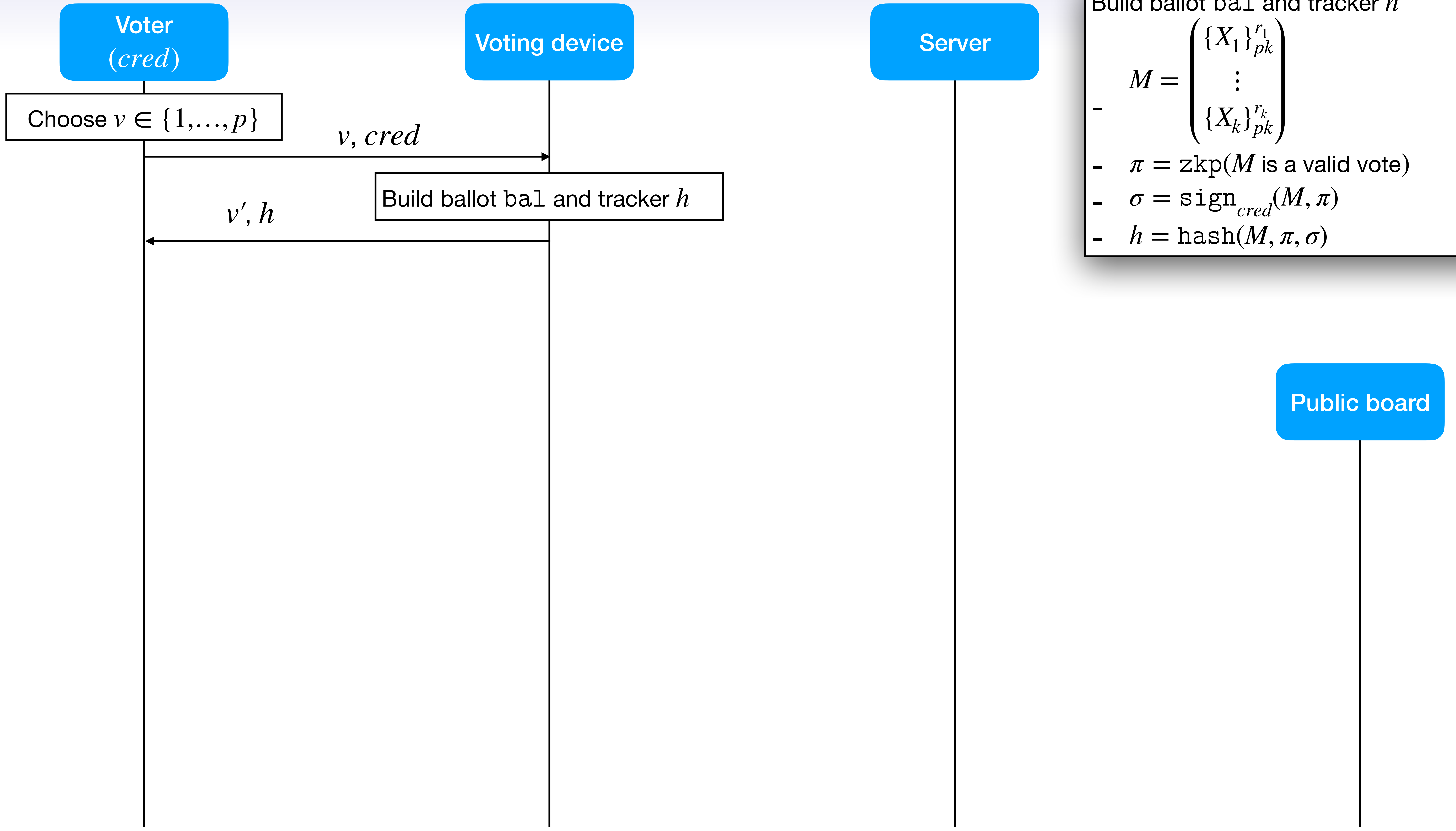
Belenios protocol



Belenios protocol



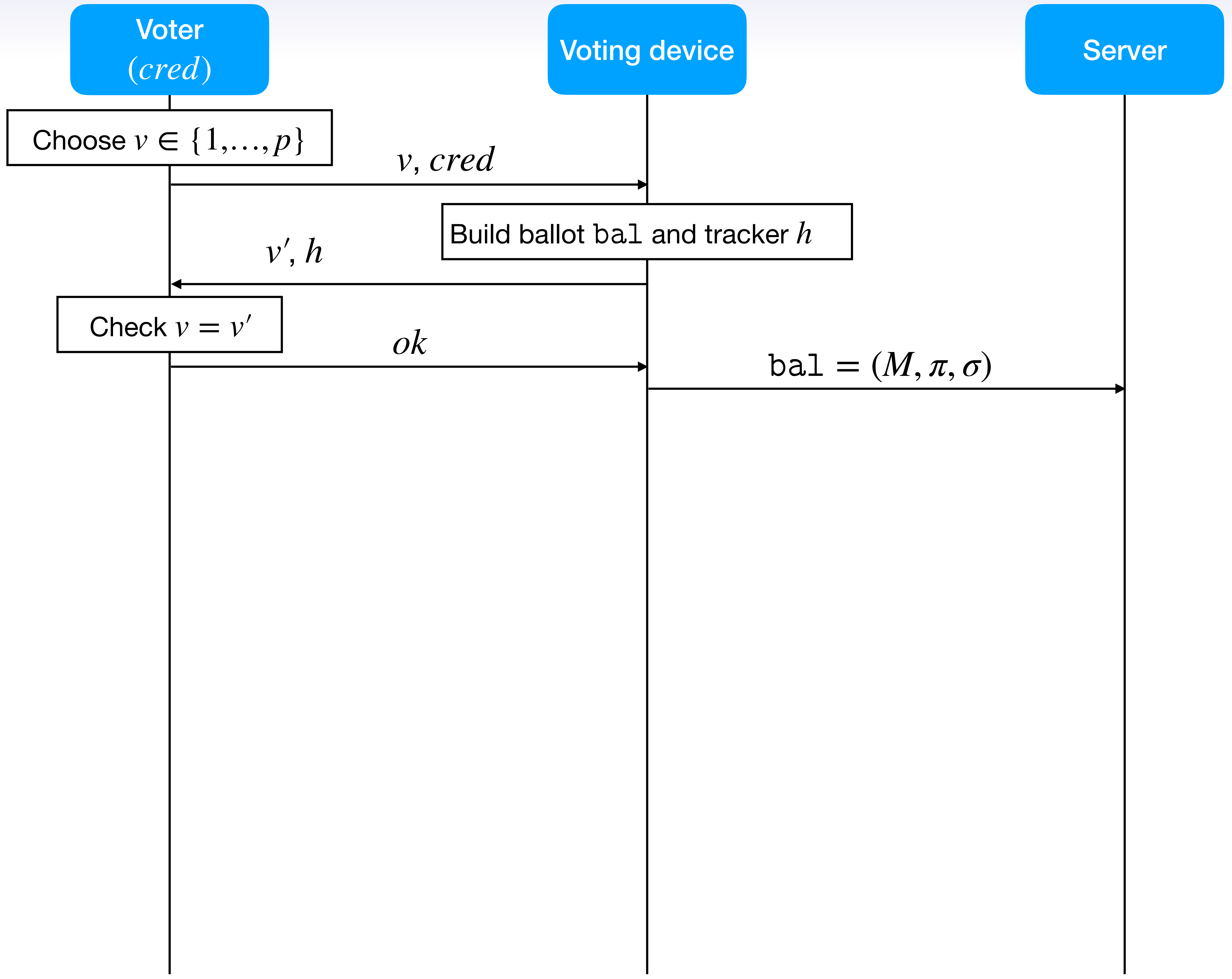
Belenios protocol


Voter
$(cred)$
Voting device
Server
Public board
Choose $v \in \{1,\ldots,p\}$
$v, cred$
Build ballot bal and tracker h
v', h

Build ballot bal and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \texttt{zkp}(M \text{ is a valid vote})$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

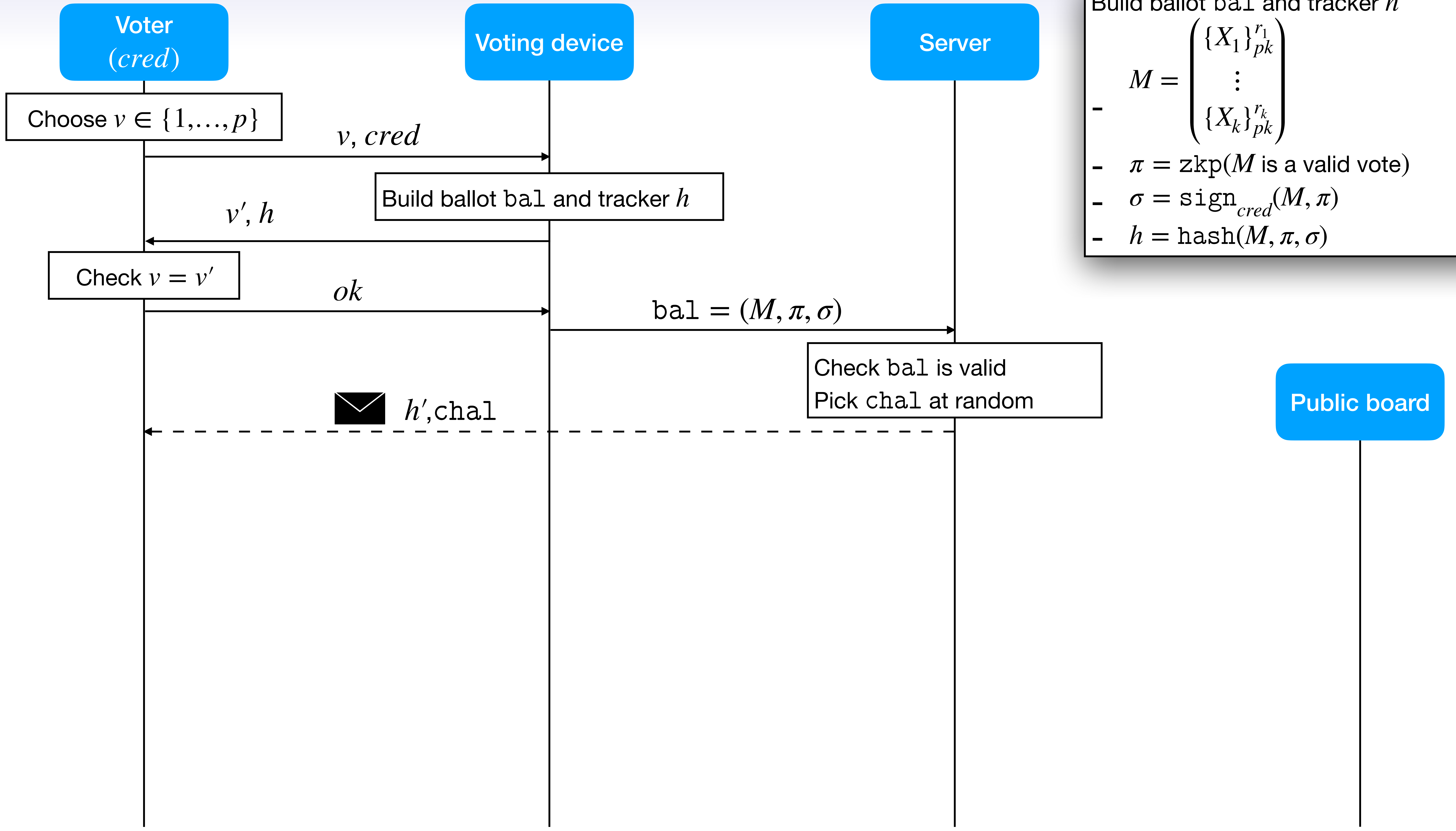
Belenios protocol


Voter
$(cred)$
Voting device
Server
Public board
Choose $v \in \{1,\ldots,p\}$
$v, cred$
Build ballot $\texttt{bal}$ and tracker h
v', h
Check $v = v'$
ok
$\texttt{bal} = (M, \pi, \sigma)$

Build ballot $\texttt{bal}$ and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \texttt{zkp}(M$ is a valid vote$)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

Belenios protocol

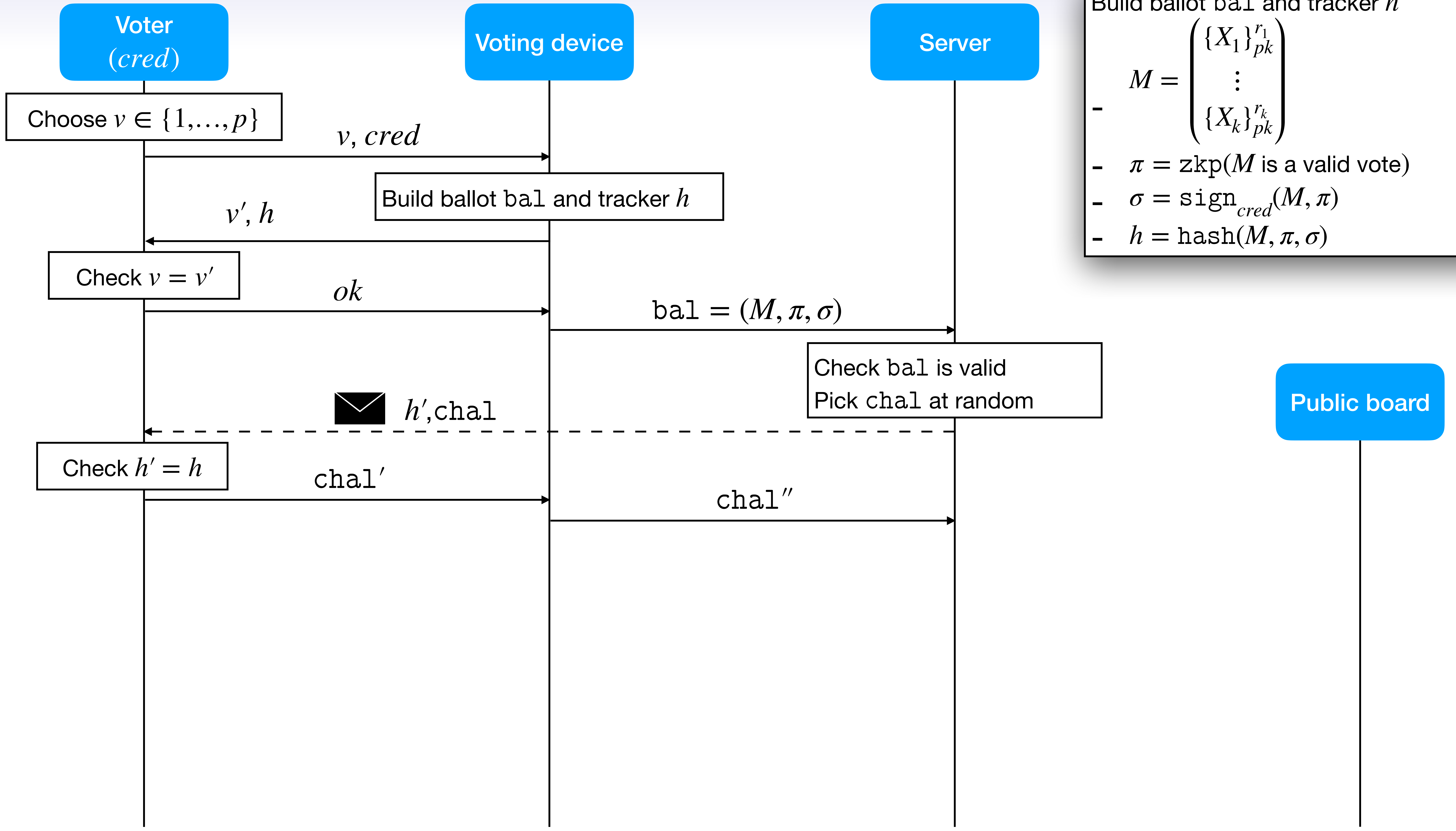


Voter
$(cred)$
Voting device
Server
Public board
Choose $v \in \{1,\ldots,p\}$
v, $cred$
Build ballot $\texttt{bal}$ and tracker h
v', h
Check $v = v'$
ok
$\texttt{bal} = (M, \pi, \sigma)$
Check $\texttt{bal}$ is valid
Pick $\texttt{chal}$ at random
h',$\texttt{chal}$

Build ballot $\texttt{bal}$ and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \texttt{zkp}(M$ is a valid vote$)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

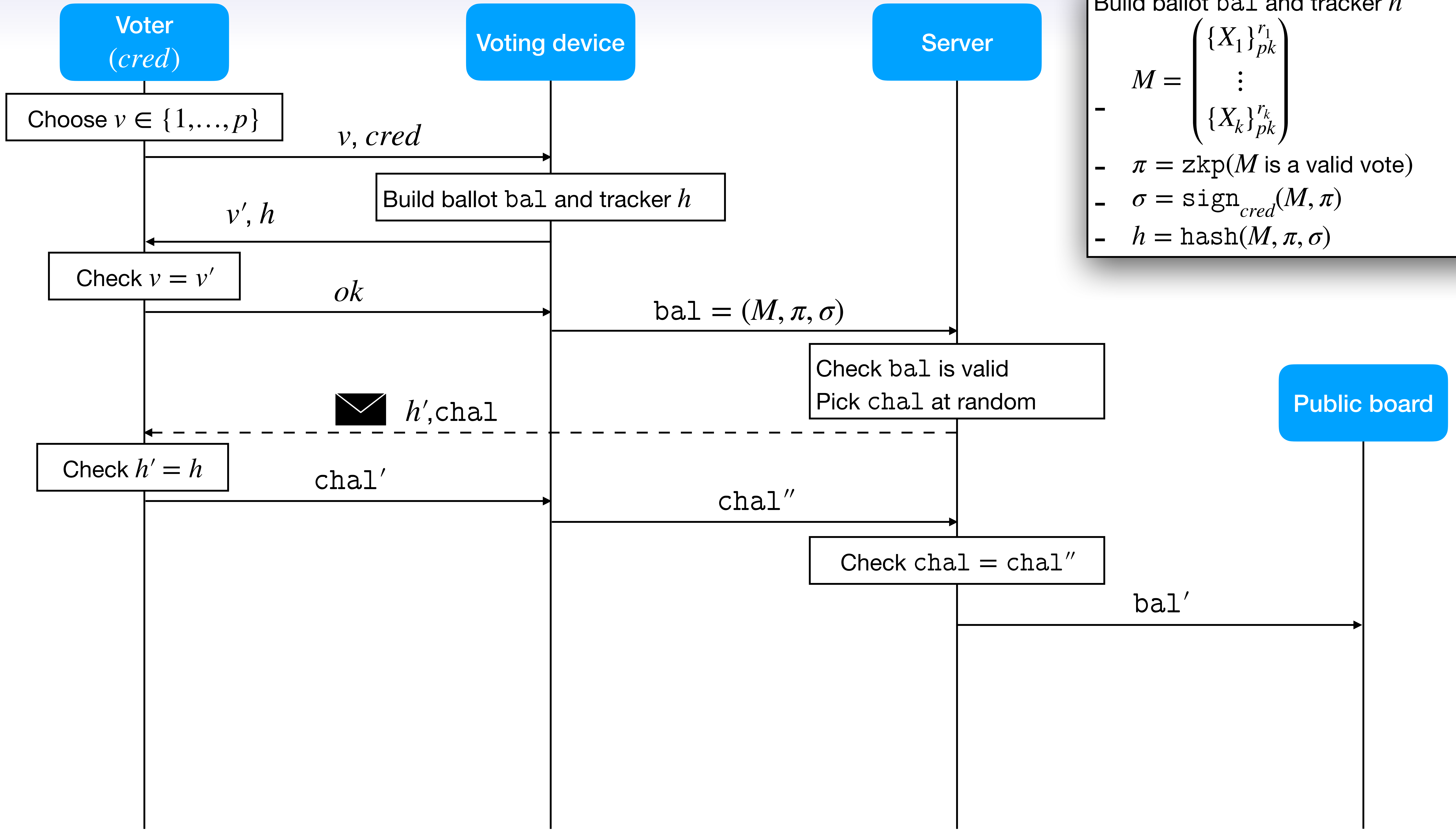
Belenios protocol



Build ballot bal and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi =$ zkp(M is a valid vote)
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Belenios protocol

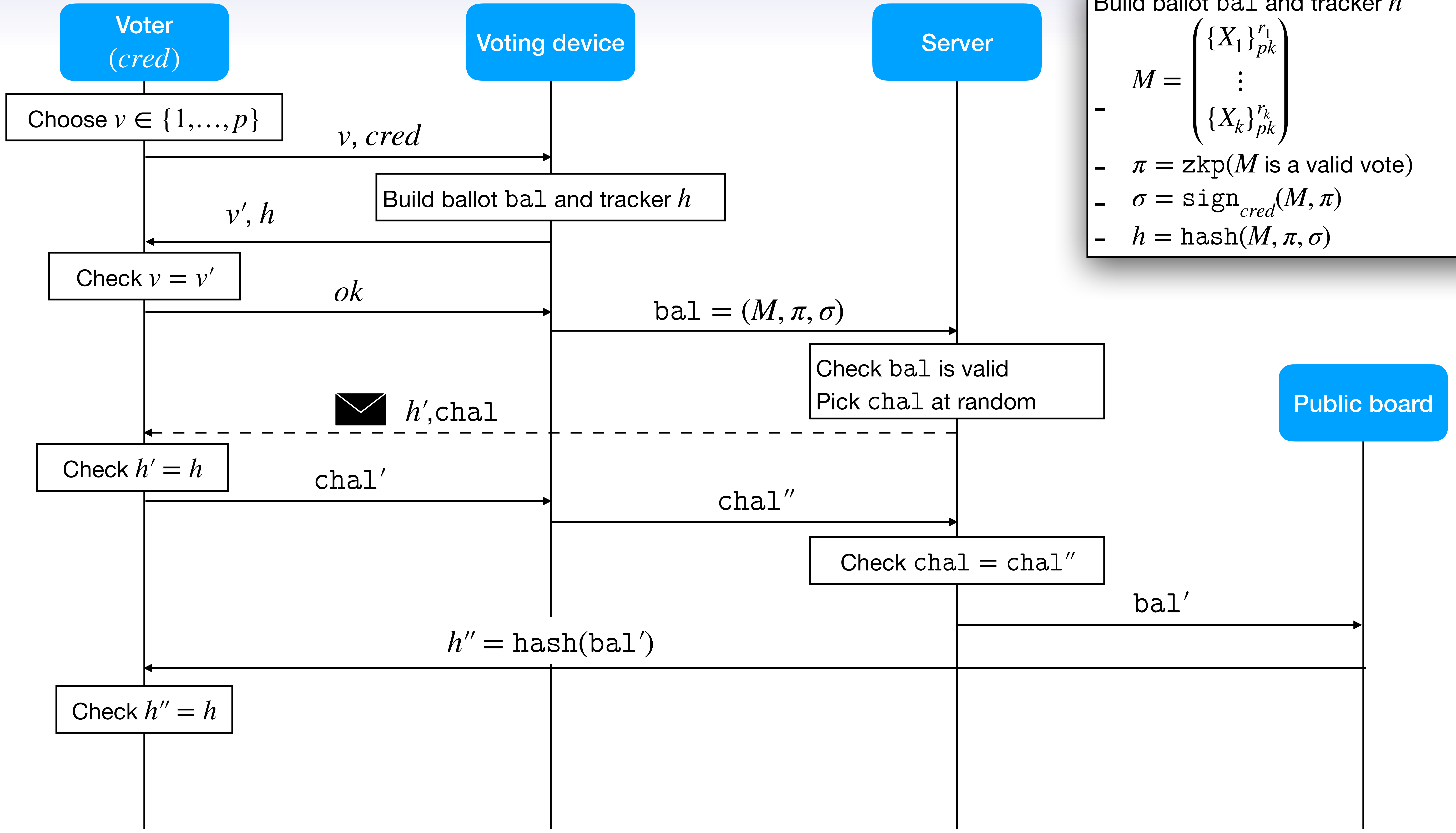


Voter
$(cred)$
Voting device
Server
Public board
Choose $v \in \{1, \ldots, p\}$
v, $cred$
Build ballot $\texttt{bal}$ and tracker h
v', h
Check $v = v'$
ok
$\texttt{bal} = (M, \pi, \sigma)$
Check $\texttt{bal}$ is valid
Pick $\texttt{chal}$ at random
h',$\texttt{chal}$
Check $h' = h$
$\texttt{chal}'$
$\texttt{chal}''$
Check $\texttt{chal} = \texttt{chal}''$
$\texttt{bal}'$

Build ballot $\texttt{bal}$ and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \texttt{zkp}(M$ is a valid vote$)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

Belenios protocol

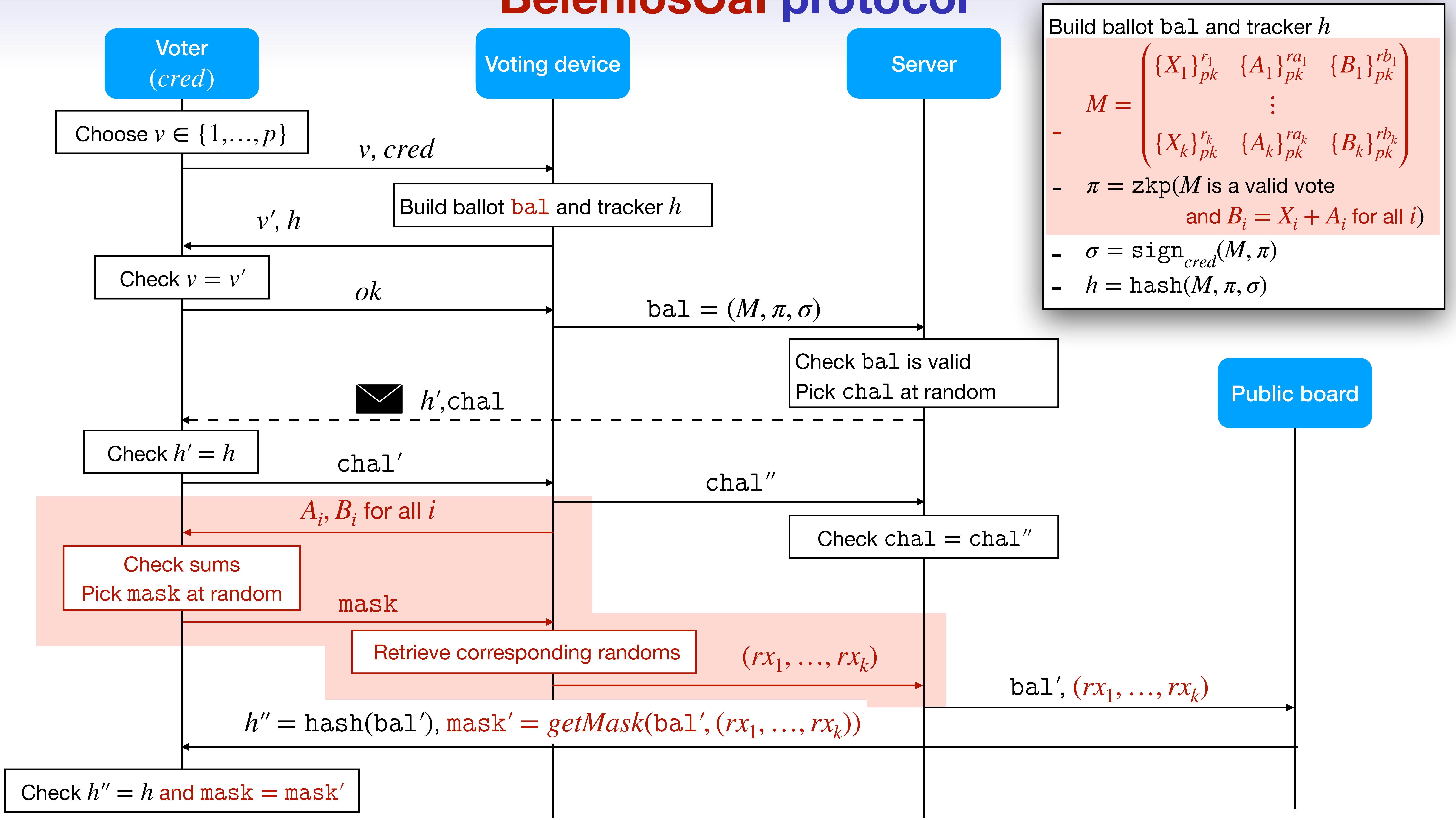


Voter
$(cred)$
Voting device
Server
Public board
Choose $v \in \{1,\dots,p\}$
v, $cred$
Build ballot $\texttt{bal}$ and tracker h
v', h
Check $v = v'$
ok
$\texttt{bal} = (M, \pi, \sigma)$
Check $\texttt{bal}$ is valid
Pick $\texttt{chal}$ at random
h', $\texttt{chal}$
Check $h' = h$
$\texttt{chal}'$
$\texttt{chal}''$
Check $\texttt{chal} = \texttt{chal}''$
$\texttt{bal}'$
$h'' = \texttt{hash}(\texttt{bal}')$
Check $h'' = h$

Build ballot $\texttt{bal}$ and tracker h

- $$M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$$
- $\pi = \texttt{zkp}(M$ is a valid vote$)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

BeleniosCal protocol

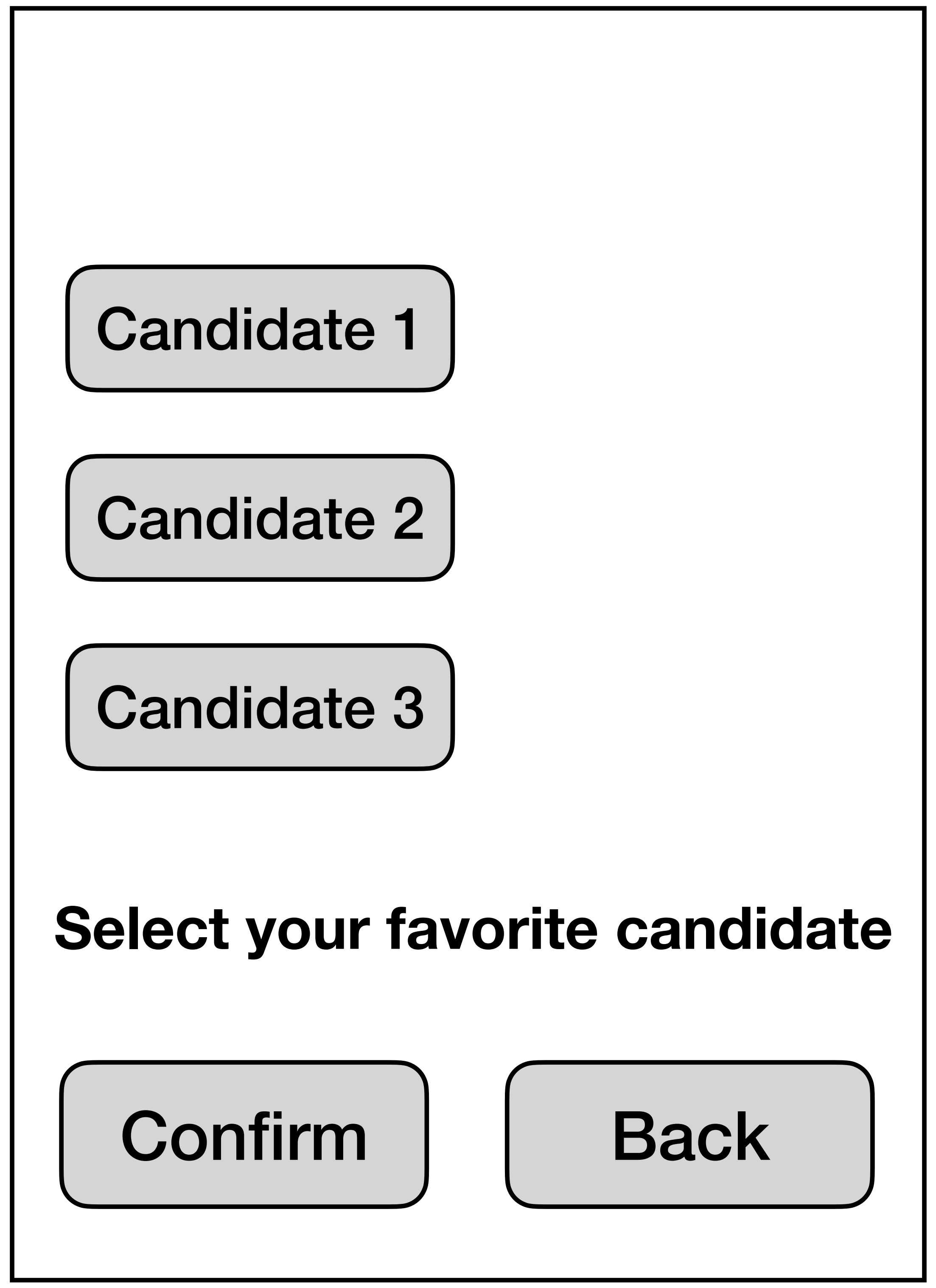


Build ballot bal and tracker h

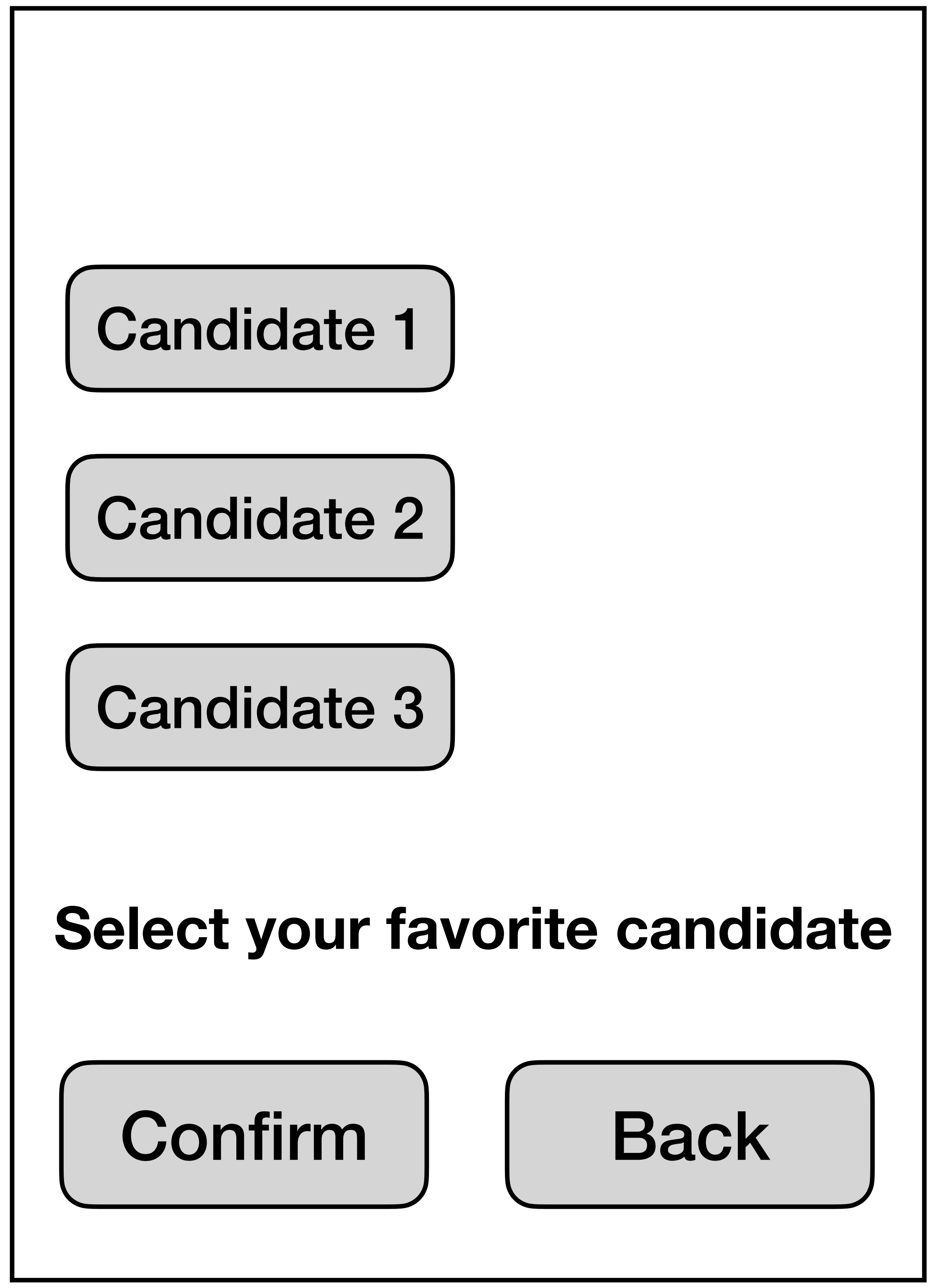
- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} & \{A_1\}_{pk}^{ra_1} & \{B_1\}_{pk}^{rb_1} \\ & \vdots & \\ \{X_k\}_{pk}^{r_k} & \{A_k\}_{pk}^{ra_k} & \{B_k\}_{pk}^{rb_k} \end{pmatrix}$
- $\pi = \mathtt{zkp}(M$ is a valid vote and $B_i = X_i + A_i$ for all $i)$
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Voter $(cred)$
Voting device
Server
Public board
Choose $v \in \{1,\ldots,p\}$
v, $cred$
Build ballot bal and tracker h
v', h
Check $v = v'$
ok
$\mathtt{bal} = (M, \pi, \sigma)$
Check bal is valid
Pick chal at random
h',chal
Check $h' = h$
$\mathtt{chal}'$
$\mathtt{chal}''$
A_i, B_i for all i
Check $\mathtt{chal} = \mathtt{chal}''$
Check sums
Pick mask at random
mask
Retrieve corresponding randoms
$(rx_1, \ldots, rx_k)$
$\mathtt{bal}', (rx_1, \ldots, rx_k)$
$h'' = \mathtt{hash}(\mathtt{bal}'), \mathtt{mask}' = getMask(\mathtt{bal}', (rx_1, \ldots, rx_k))$
Check $h'' = h$ and $\mathtt{mask} = \mathtt{mask}'$

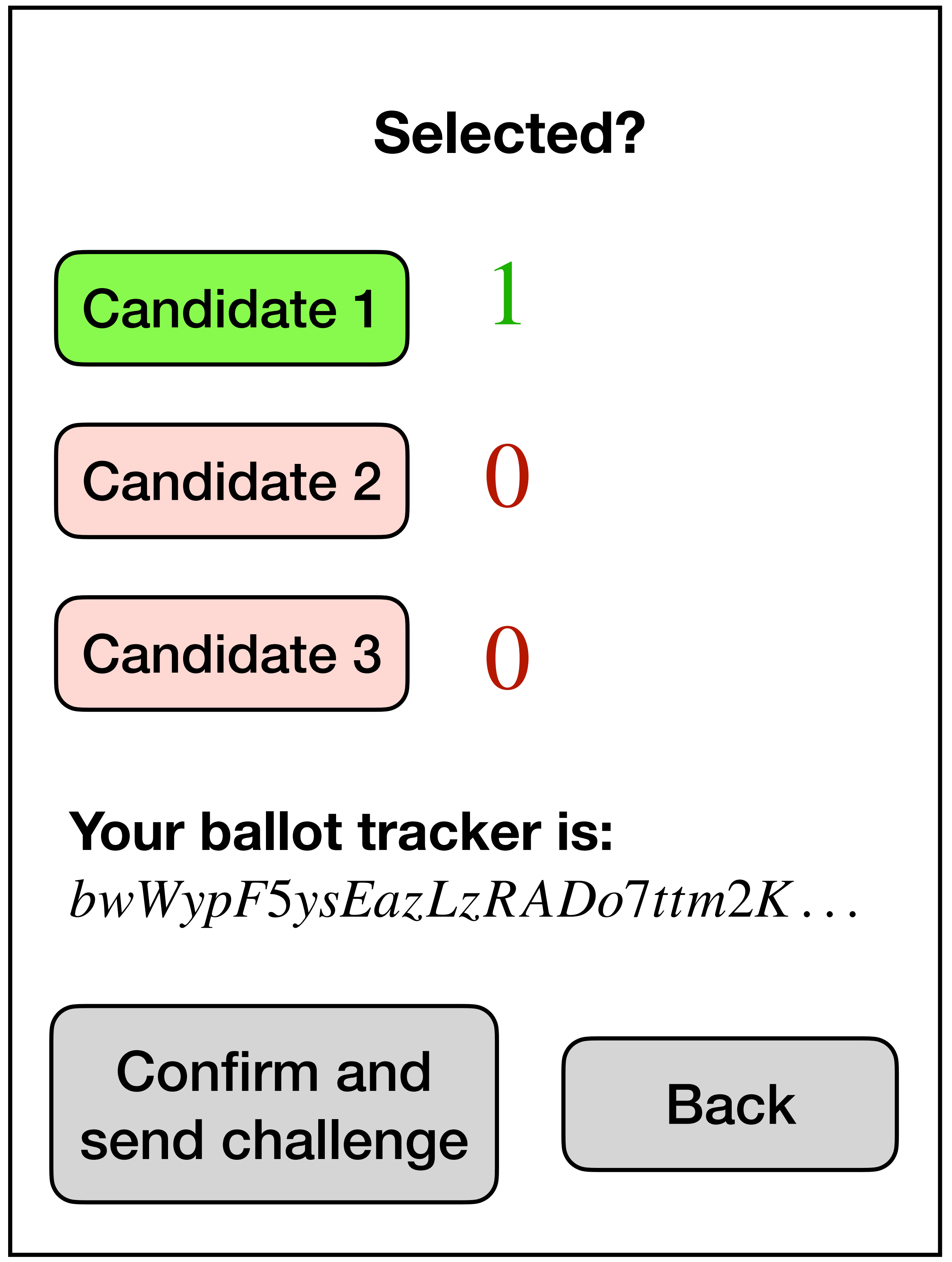
User experience



User experience



Select your favorite candidate

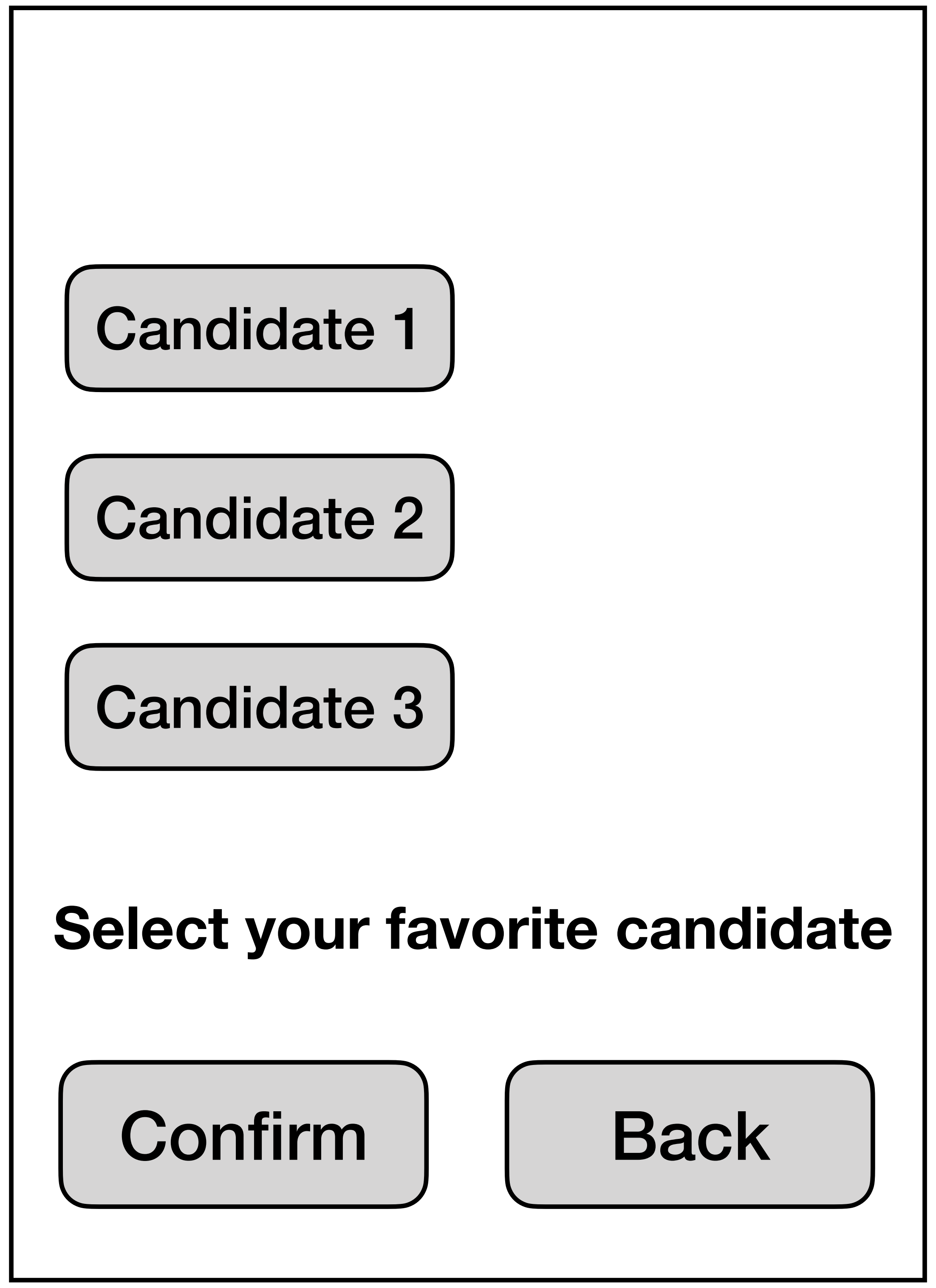


Selected?

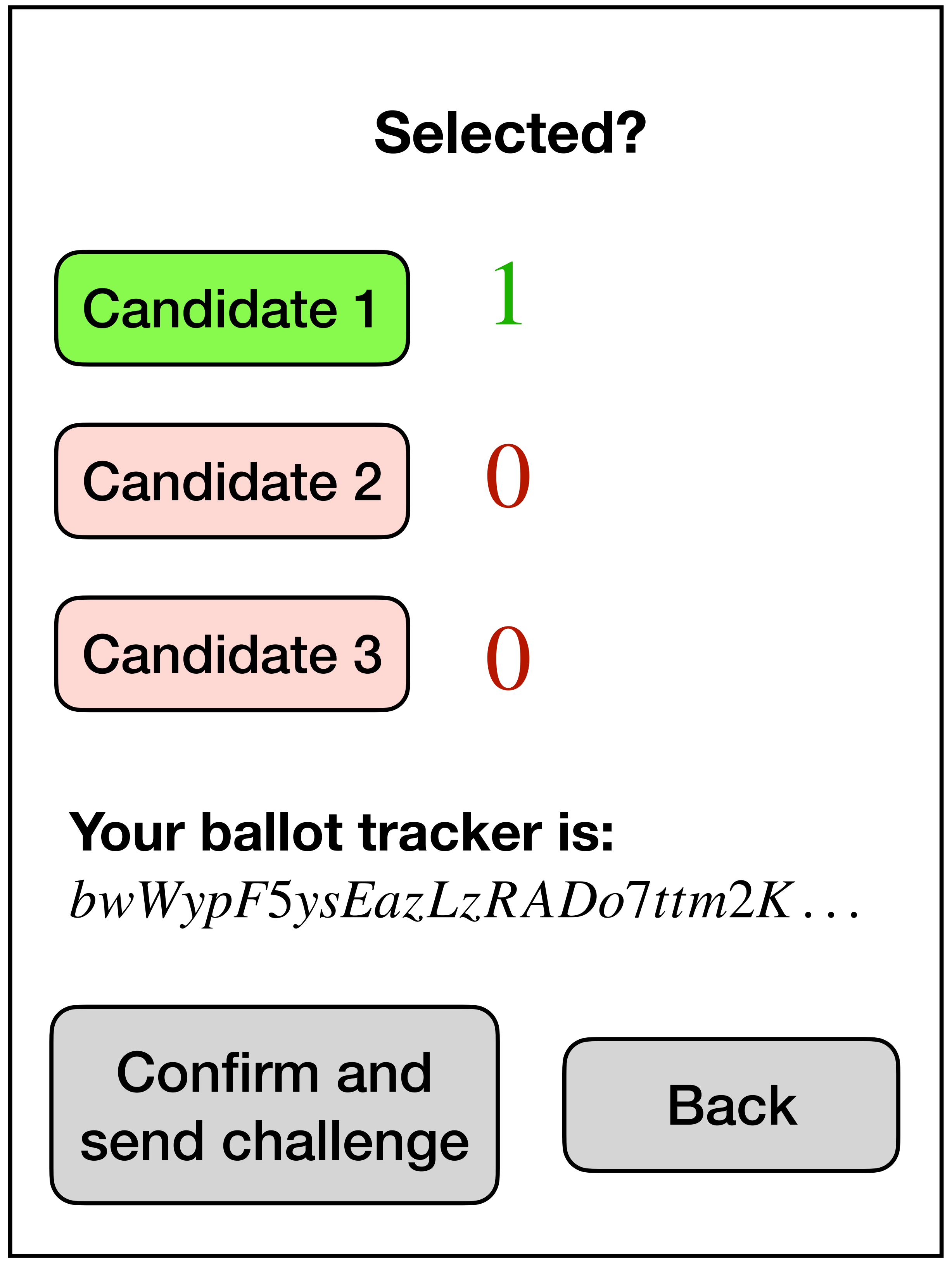
Candidate 1	1
Candidate 2	0
Candidate 3	0

Your ballot tracker is:
$bwWypF5ysEazLzRADo7ttm2K\ldots$

User experience

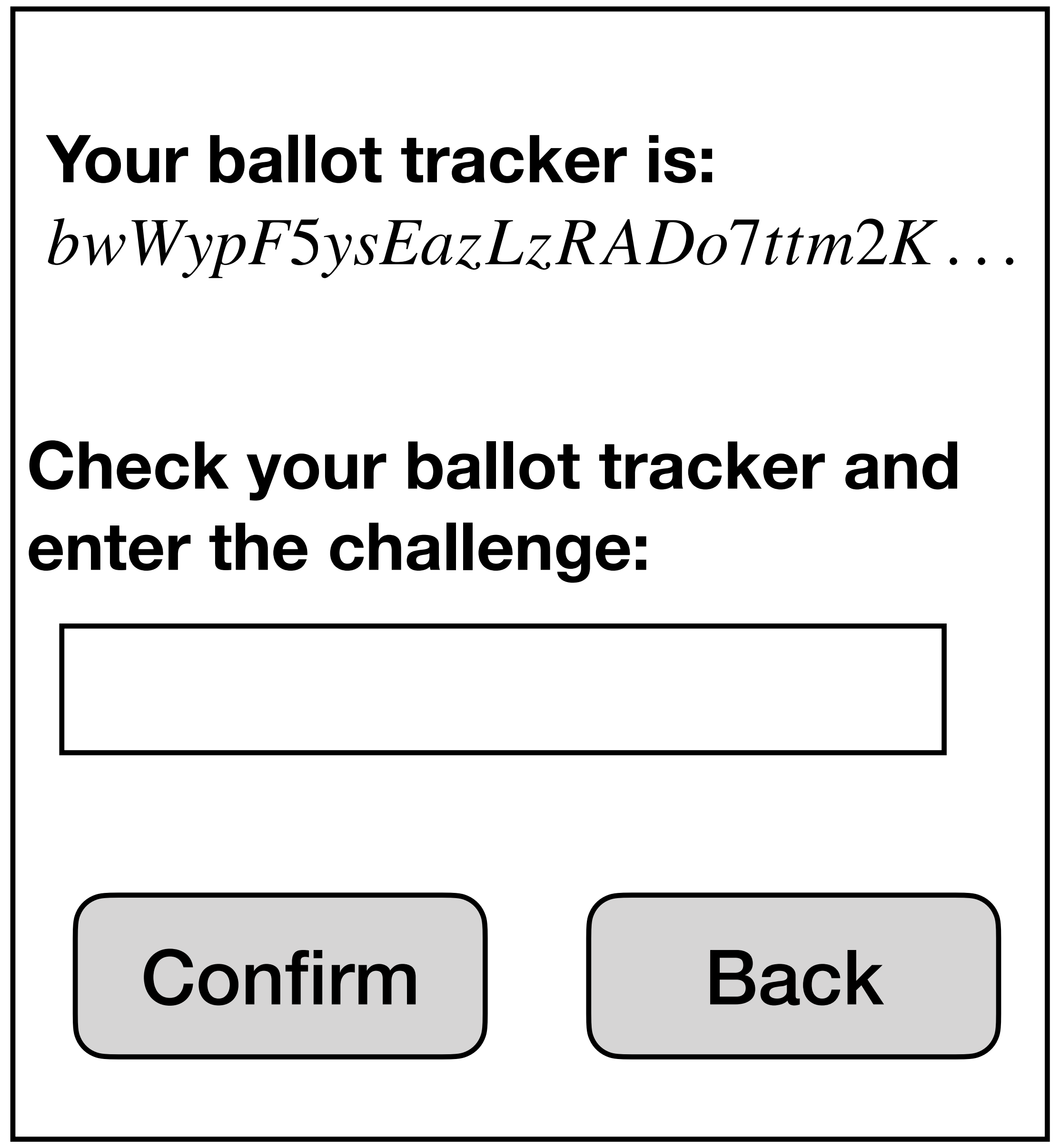


Select your favorite candidate



Selected?

Your ballot tracker is:
bwWypF5ysEazLzRADo7ttm2K...



Your ballot tracker is:
bwWypF5ysEazLzRADo7ttm2K...

Check your ballot tracker and enter the challenge:

User experience

Candidate 1
Candidate 2
Candidate 3
Confirm
Back
Selected?
Candidate 1
Candidate 2
Candidate 3
Confirm and send challenge
Back
Confirm
Back
Selected?
Audit codes
Candidate 1
Candidate 2
Candidate 3
Cast ballot
Back

Select your favorite candidate

Selected?

Candidate 1	1
Candidate 2	0
Candidate 3	0

Your ballot tracker is:
bwWypF5ysEazLzRADo7ttm2K...

Your ballot tracker is:
bwWypF5ysEazLzRADo7ttm2K...

Check your ballot tracker and enter the challenge:

	Selected?	Audit codes
Candidate 1	$1 + 9 = 10$	
Candidate 2	$0 + 3 = 3$	
Candidate 3	$0 + 6 = 6$	

Check that the additions are correct and select one audit code per line

User experience

Candidate 1
Candidate 2
Candidate 3
Confirm
Back
Selected?
Candidate 1
Candidate 2
Candidate 3
Confirm and send challenge
Back
Confirm
Back
Selected?
Audit codes
Candidate 1
Candidate 2
Candidate 3
Cast ballot
Back

Select your favorite candidate

Selected?

Candidate 1 1

Candidate 2 0

Candidate 3 0

Your ballot tracker is: *bwWypF5ysEazLzRADo7ttm2K...*

Your ballot tracker is: *bwWypF5ysEazLzRADo7ttm2K...*

Check your ballot tracker and enter the challenge:

Candidate 1 $1 + \boxed{9} = 10$

Candidate 2 $0 + 3 = \boxed{3}$

Candidate 3 $0 + 6 = \boxed{6}$

Check that the additions are correct and select one audit code per line

User experience

Candidate 1
Candidate 2
Candidate 3
Confirm
Back
Selected?
Candidate 1
Candidate 2
Candidate 3
Confirm and send challeng
Bulletin board
Candidate 1 – + 9 = –
Candidate 2 – + – = 3
Candidate 3 – + – = 6
Selected?
Audit codes
Candidate 1 1 + 9 = 10
Candidate 2 0 + 3 = 3
Candidate 3 0 + 6 = 6
Cast ballot
Back

Select your favorite candidate

Your ballot trac
bwWypF5ysEazL

Ballot tracker:
bwWypF5ysEazLzRADo7ttm2K...

Check that the additions are correct and select one audit code per line

Computation cost

Belenios ballot

Builds ballot $\texttt{bal}$ and tracker h

- $$M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$$
- $\pi = \texttt{zkp}(M$ is a valid vote$)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

BeleniosCal ballot

Builds ballot $\texttt{bal}$ and tracker h

- $$M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} & \{A_1\}_{pk}^{ra_1} & \{B_1\}_{pk}^{rb_1} \\ & \vdots & \\ \{X_k\}_{pk}^{r_k} & \{A_k\}_{pk}^{ra_k} & \{B_k\}_{pk}^{rb_k} \end{pmatrix}$$
- $\pi = \texttt{zkp}(M$ is a valid vote
  and $B_i = X_i + A_i$ for all $i)$
- $\sigma = \texttt{sign}_{cred}(M, \pi)$
- $h = \texttt{hash}(M, \pi, \sigma)$

Computation cost

Belenios ballot

Builds ballot `bal` and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \mathtt{zkp}(M$ is a valid vote$)$
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Ciphertexts: 1 ElGamal encryption per candidate

ZKP:

- $X_i \in \{0,1\} \Rightarrow$ 5 exponentiations per candidate
- the voter chooses a valid combination $\Rightarrow$ it depends

Signature: 1 signature

BeleniosCal ballot

Builds ballot `bal` and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} & \{A_1\}_{pk}^{ra_1} & \{B_1\}_{pk}^{rb_1} \\ & \vdots & \\ \{X_k\}_{pk}^{r_k} & \{A_k\}_{pk}^{ra_k} & \{B_k\}_{pk}^{rb_k} \end{pmatrix}$
- $\pi = \mathtt{zkp}(M$ is a valid vote and $B_i = X_i + A_i$ for all $i)$
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Ciphertexts: $1{+}2$ ElGamal encryptions per candidate

ZKP:

- $X_i \in \{0,1\} \Rightarrow$ 5 exponentiations per candidate
- the voter chooses a valid combination $\Rightarrow$ it depends
- $B_i = X_i + A_i \mod n \Rightarrow$ 5 exponentiations per candidate

Signature: 1 signature

Computation cost

Belenios ballot

Builds ballot bal and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$
- $\pi = \mathtt{zkp}(M$ is a valid vote)
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Ciphertexts: 1 ElGamal encryption per candidate

ZKP:

- $X_i \in \{0,1\} \Rightarrow$ 5 exponentiations per candidate
- the voter chooses a valid combination $\Rightarrow$ it depends

Signature: 1 signature

A BeleniosCal ballot is (at most) 3 times more expensive

BeleniosCal ballot

Builds ballot bal and tracker h

- $M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} & \{A_1\}_{pk}^{ra_1} & \{B_1\}_{pk}^{rb_1} \\ & \vdots & \\ \{X_k\}_{pk}^{r_k} & \{A_k\}_{pk}^{ra_k} & \{B_k\}_{pk}^{rb_k} \end{pmatrix}$
- $\pi = \mathtt{zkp}(M$ is a valid vote and $B_i = X_i + A_i$ for all $i)$
- $\sigma = \mathtt{sign}_{cred}(M, \pi)$
- $h = \mathtt{hash}(M, \pi, \sigma)$

Ciphertexts: $1+2$ ElGamal encryptions per candidate

ZKP:

- $X_i \in \{0,1\} \Rightarrow$ 5 exponentiations per candidate
- the voter chooses a valid combination $\Rightarrow$ it depends
- $B_i = X_i + A_i \mod n \Rightarrow$ 5 exponentiations per candidate

Signature: 1 signature

Security analysis

ProVerif

- An automatic prover for symbolic analysis
- Handle trace-based properties for verifiability
- Handle equivalence-based properties for vote secrecy

Security analysis

ProVerif

- An automatic prover for symbolic analysis
- Handle trace-based properties for verifiability
- Handle equivalence-based properties for vote secrecy

2 main challenges

- Probabilities: ProVerif does not support probabilistic choices

  The voter receives a commitment on their ballot before doing their choice

  ➡ It is enough to model that the voter could audit both codes

Security analysis

ProVerif

- An automatic prover for symbolic analysis
- Handle trace-based properties for verifiability
- Handle equivalence-based properties for vote secrecy

2 main challenges

- Probabilities: ProVerif does not support probabilistic choices

  The voter receives a commitment on their ballot before doing their choice

  ➡ It is enough to model that the voter could audit both codes

- Additions: ProVerif does not support arithmetics in $\mathbb{Z}_n$

  ➡ reachability: over-approximate the "+" operator

  ➡ equivalence: prove a relation preservation

[Cortier et al - 2022]

Modeling arithmetics in $\mathbb{Z}_n$

Modeling:

- integers are modeled by abstract atomic values, $x, y, a, b, c, \ldots$
- whenever someone checks $b \overset{?}{=} x + a$, we execute the event $isSum(x, a, b)$

Modeling arithmetics in $\mathbb{Z}_n$

Modeling:

- integers are modeled by abstract atomic values, $x, y, a, b, c, \ldots$
- whenever someone checks $b \stackrel{?}{=} x + a$, we execute the event $isSum(x, a, b)$

Reachability properties:

« For all $x, a \in \mathbb{Z}_n$, there exists a unique $b \in \mathbb{Z}_n$ such that $b = x + a$ »

$\Downarrow$

Restrictions such that

$$isSum(x, a, b) \ \wedge \ isSum(x, a, b') \Rightarrow b = b'$$

$$isSum(x, a, b) \ \wedge \ isSum(x, a', b) \Rightarrow a = a'$$

$$\ldots$$

Modeling arithmetics in $\mathbb{Z}_n$

Modeling:

- integers are modeled by abstract atomic values, $x, y, a, b, c, \ldots$
- whenever someone checks $b \overset{?}{=} x + a$, we execute the event $isSum(x, a, b)$

Reachability properties:

« For all $x, a \in \mathbb{Z}_n$, there exists a unique $b \in \mathbb{Z}_n$ such that $b = x + a$ »

Restrictions such that

$$isSum(x,a,b) \;\wedge\; isSum(x,a,b') \Rightarrow b = b'$$

$$isSum(x,a,b) \;\wedge\; isSum(x,a',b) \Rightarrow a = a'$$

$$\ldots$$

Equivalence properties: relation preservation

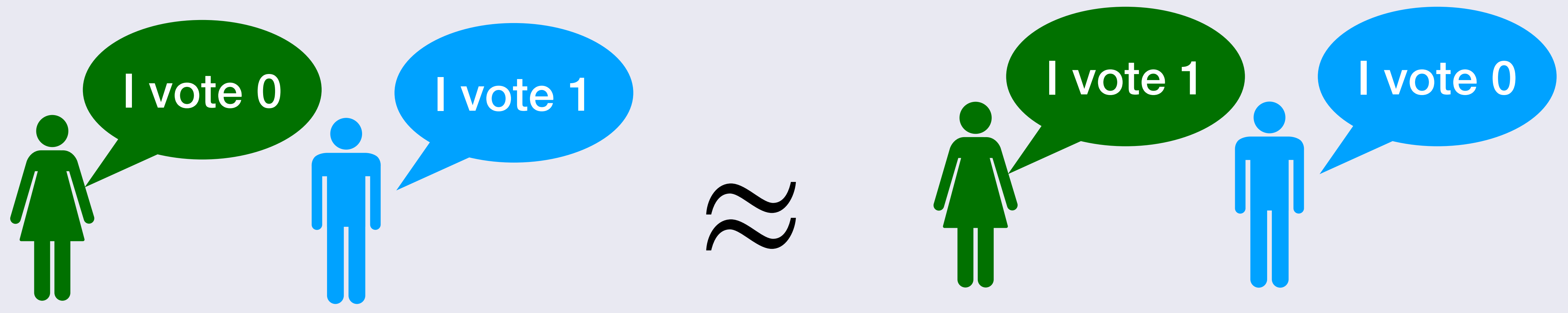
Lemma (intuition): given two processes P and Q, for all traces $tr_P \in Traces(P)$ and $tr_Q \in Traces(Q)$ such that $tr_P \approx tr_Q$ we have:

$$isSum(x,a,b) \in tr_P \;\Leftrightarrow\; isSum(x,a,b) \in tr_Q$$

(related to the notion of bi-process and diff-equivalence)

Security properties

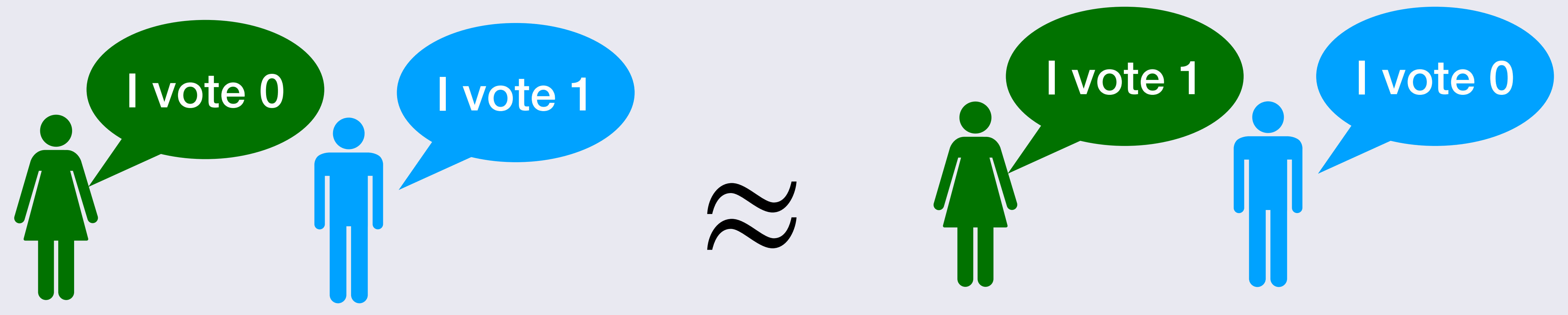
Vote secrecy
[Kremer *et al* - 2009]



Security properties

Vote secrecy

[Kremer *et al* - 2009]



Verifiability

- **Cast-as-intended:**

  $\texttt{Verified}(\mathsf{LR}, id, h, v) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{onBoard}(vk', b, h, r, \mathsf{X}) \wedge (b \text{ encrypts candidate } v)$

- **No clash attack:**

  $\texttt{Verified}(\mathsf{L}, id, h, v) \wedge \texttt{Verified}(\mathsf{R}, id', h, v') \Rightarrow \mathit{false}$

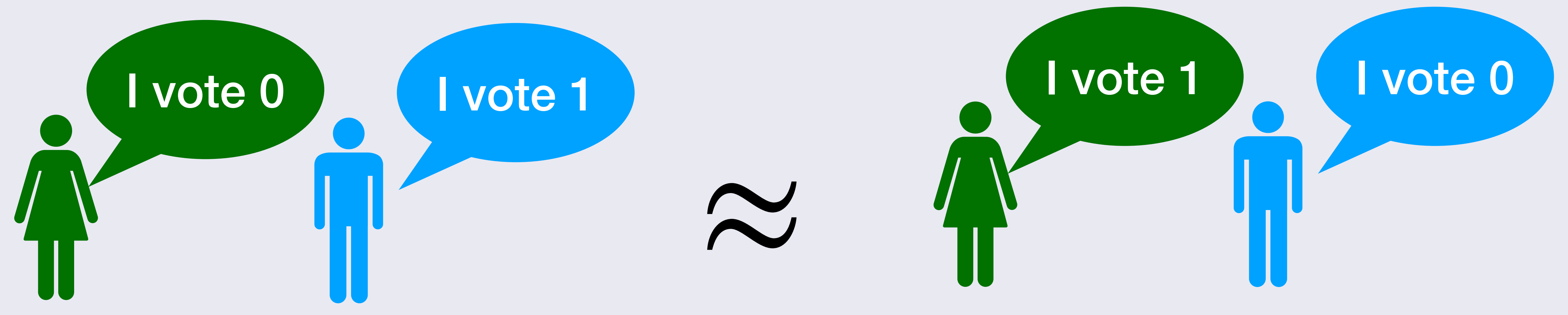
- **Recorded-as-cast:**

  (strong) $\texttt{onBoard}(vk, b, h, r, \mathsf{X}) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{Voted}(id', vk, h)$

  (standard) $\texttt{onBoard}(vk, b, h, r, \mathsf{X}) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{HasInitiatedVote}(id', vk)$

Security properties

Vote secrecy
[Kremer *et al* - 2009]



Verifiability

- **Cast-as-intended:**

  $\texttt{Verified}(\mathsf{LR}, id, h, v) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{onBoard}(vk', b, h, r, \mathsf{X}) \wedge (b \text{ encrypts candidate } v)$

- **No clash attack:**

  $\texttt{Verified}(\mathsf{L}, id, h, v) \wedge \texttt{Verified}(\mathsf{R}, id', h, v') \Rightarrow \mathit{false}$

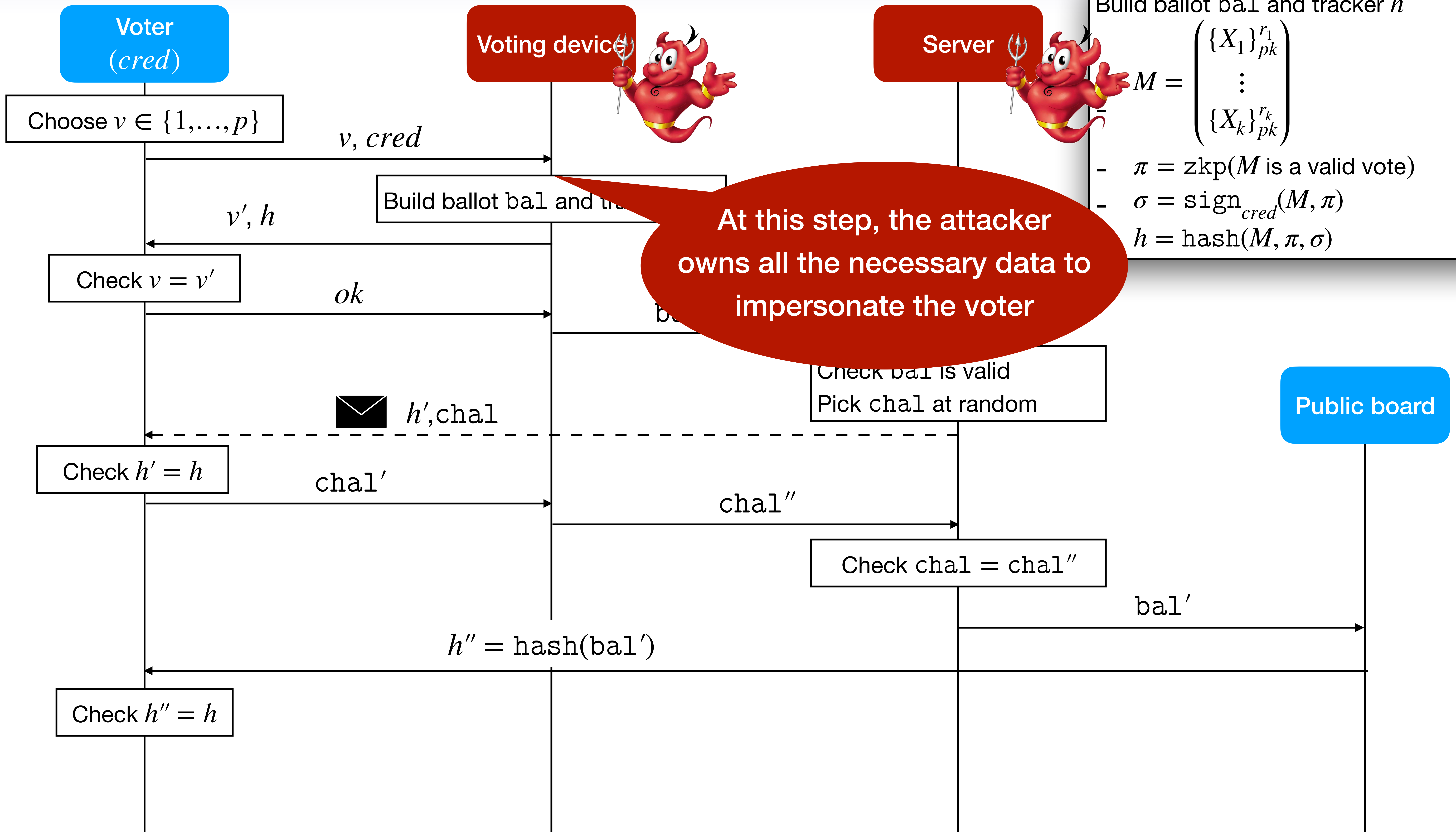
  A clash is possible when auditing on the same side…
  detected with probability $1/2$

- **Recorded-as-cast:**

  (strong) $\texttt{onBoard}(vk, b, h, r, \mathsf{X}) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{Voted}(id', vk, h)$

  (standard) $\texttt{onBoard}(vk, b, h, r, \mathsf{X}) \wedge \texttt{Honest}(id, vk) \Rightarrow \texttt{HasInitiatedVote}(id', vk)$

Recorded-as-cast: strong vs standard



Build ballot bal and tracker h

$$M = \begin{pmatrix} \{X_1\}_{pk}^{r_1} \\ \vdots \\ \{X_k\}_{pk}^{r_k} \end{pmatrix}$$

- $\pi = \text{zkp}(M$ is a valid vote$)$
- $\sigma = \text{sign}_{cred}(M, \pi)$

$h = \text{hash}(M, \pi, \sigma)$

Results

	Voter	Voting client	Voting server	Registrar	Trustees
Cast-as-intended	😇	👺	👺	👺	👺
No clash	😇	👺	👺	👺	👺
Recorded-as-cast (strong)	😇	😇	👺	😇	👺
	😇	👺	😇	👺	👺
Recorded-as-cast (standard)	😇	👺	👺	😇	👺
	😇	👺	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Results

	Voter	Voting client	Voting server	Registrar	Trustees
Cast-as-intended	😇	👺	👺	👺	👺
No clash	😇	👺	👺	👺	👺
Recorded-as-cast (strong)	😇	😇	👺	😇	👺
	😇	👺	😇	👺	👺
Recorded-as-cast (standard)	😇	👺	👺	😇	👺
	😇	👺	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Results

	Voter	Voting client	Voting server	Registrar	Trustees
Cast-as-intended	😇	👺	👺	👺	👺
No clash	😇	👺	👺	👺	👺
Recorded-as-cast (strong)	😇	😇	👺	😇	👺
	😇	👺	😇	👺	👺
Recorded-as-cast (standard)	😇	👺	👺	😇	👺
	😇	👺	😇	👺	👺
Vote secrecy	😇	😇	👺	👺	😇 (k out of n)

Summary

		Voter	Voting client	Voting server	Registrar	Trustees
Verifiability	(standard only)	😇	👺	👺	😇	👺
	(standard and strong)	😇	👺	😇	👺	👺
Vote secrecy		😇	😇	👺	👺	😇 (k out of n)

BeleniosCal

- extends Belenios with cast-as-intended
- preserves Belenios vote secrecy guarantees
- is formally proven secure
- still does not require expensive computations

Future work

Implement the protocol in Belenios framework

propose it as a new feature

Evaluate its usability in practice

understandability

UX (representation of additions in $\mathbb{Z}_n$, choice of n, etc)

Evaluate its acceptability

Do people understand and accept “probabilistic security”?