Verifying complex ballots with a single (constant size) verification code

Rui Joaquim

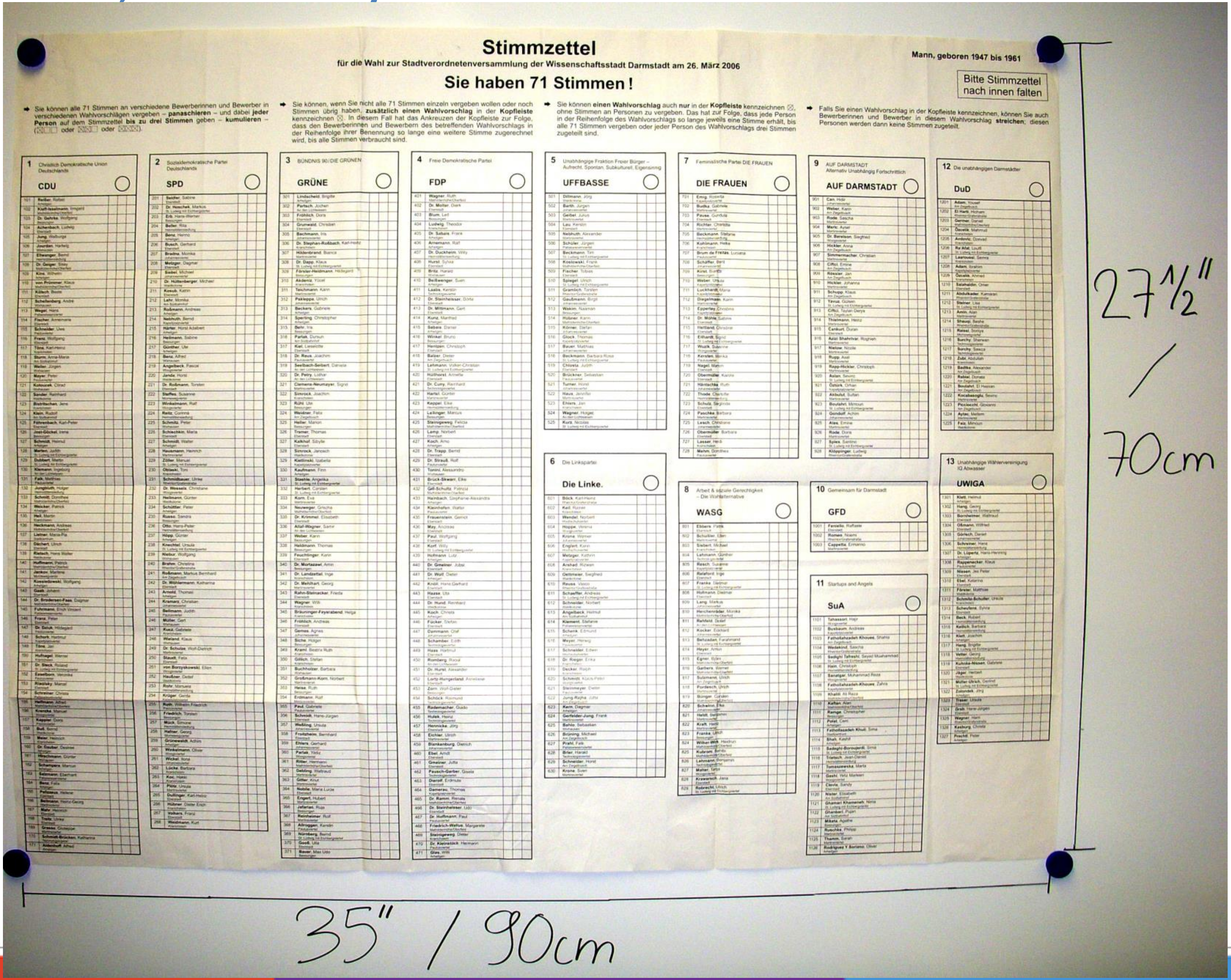
rui.joaquim@uni.lu

e-Voting PhD days, 14-15 November 2013, Muenchenwiler

Introduction

- The verification of a vote encryption is needed to:
  - Guarantee that a valid vote is encrypted.
    - Important for both mix-net and homomorphic tallying.
  - Guarantee that the voter’s choice is encrypted correctly.
    - Important for cast-as-intended and end-to-end verifiability.
- A complex ballot is one that has a large number of valid vote possibilities.

Complex ballot example 1
Darmstadt, Germany



Stimmzettel

für die Wahl zur Stadtverordnetenversammlung der Wissenschaftsstadt Darmstadt am 26. März 2006

Sie haben 71 Stimmen!

Mann, geboren 1947 bis 1961

Bitte Stimmzettel nach innen falten

- Sie können alle 71 Stimmen an verschiedene Bewerberinnen und Bewerber in verschiedenen Wahlvorschlägen vergeben – **panaschieren** – und dabei **jeder Person** auf dem Stimmzettel **bis zu drei Stimmen** geben – **kumulieren** – (☒☐☐ oder ☒☒☐ oder ☒☒☒).
- Sie können, wenn Sie nicht alle 71 Stimmen einzeln vergeben wollen oder noch Stimmen übrig haben, **zusätzlich einen Wahlvorschlag** in der **Kopfleiste** kennzeichnen ☒. In diesem Fall hat das Ankreuzen der Kopfleiste zur Folge, dass den Bewerberinnen und Bewerbern des betreffenden Wahlvorschlags in der Reihenfolge ihrer Benennung so lange eine weitere Stimme zugerechnet wird, bis alle Stimmen verbraucht sind.
- Sie können **einen Wahlvorschlag** auch **nur** in der **Kopfleiste** kennzeichnen ☒, ohne Stimmen an Personen zu vergeben. Das hat zur Folge, dass jede Person in der Reihenfolge des Wahlvorschlags so lange jeweils eine Stimme erhält, bis alle 71 Stimmen vergeben oder jeder Person des Wahlvorschlags drei Stimmen zugeteilt sind.
- Falls Sie einen Wahlvorschlag in der Kopfleiste kennzeichnen, können Sie auch Bewerberinnen und Bewerber in diesem Wahlvorschlag **streichen**; diesen Personen werden dann keine Stimmen zugeteilt.

1 Christlich Demokratische Union Deutschlands **CDU** ○

2 Sozialdemokratische Partei Deutschlands **SPD** ○

3 BÜNDNIS 90/DIE GRÜNEN **GRÜNE** ○

4 Freie Demokratische Partei **FDP** ○

5 Unabhängige Fraktion Freier Bürger – Aufrecht, Spontan, Subkulturell, Eigensinnig **UFFBASSE** ○

6 Die Linkspartei **Die Linke.** ○

7 Feministische Partei DIE FRAUEN **DIE FRAUEN** ○

8 Arbeit & soziale Gerechtigkeit – Die Wahlalternative **WASG** ○

9 AUF DARMSTADT Alternativ Unabhängig Fortschrittlich **AUF DARMSTADT** ○

10 Gemeinsam für Darmstadt **GFD** ○

11 Startups und Angels **SuA** ○

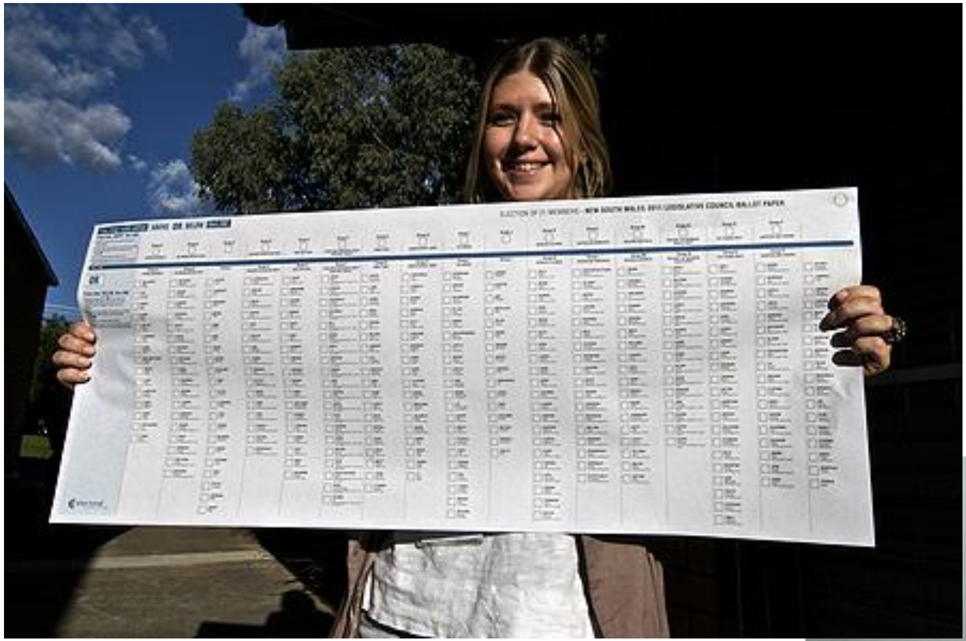
12 Die unabhängigen Darmstädter **DuD** ○

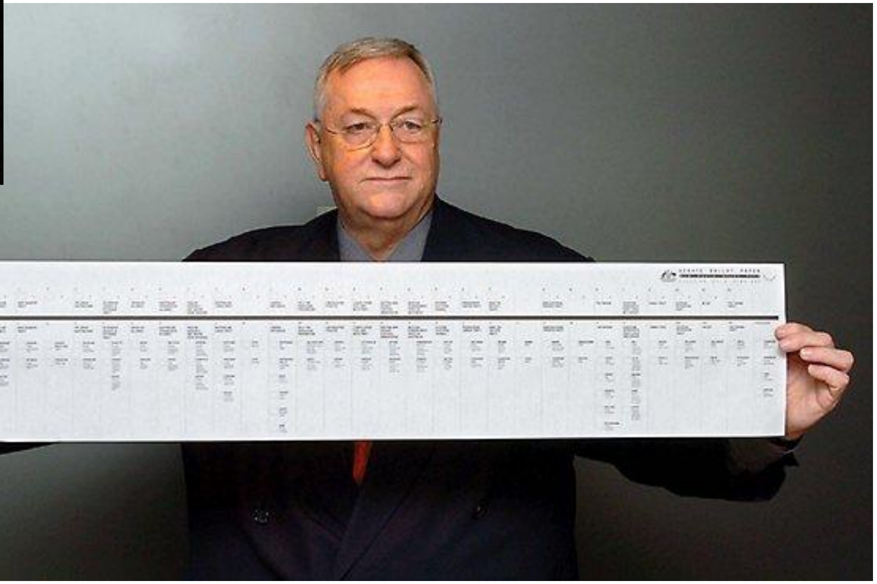
13 Unabhängige Wählervereinigung IG Abwasser **UWIGA** ○

27½" / 70cm

35" / 90cm

Complex ballot example 2
Australia





Related work

	This work		Groth 2005		Helios	
	P	V	P	V	P	V
Approval K-out-of-N	2N [+ 3N]	4N [+ 5N]	6K + 4	3K + 3	6N + 2	8N + 4
(0-N)-out-of-N	4N [+ 3N]	8N [+ 5N]	2N + 4	N + 3	6N	8N
(K_{min}-K_{max})-out-of-N	2(N + K_{max}-K_{min}) [+ 3N]	4(N + K_{max}-K_{min}) [+ 5N]	-	-	6N + 4(K_{max}-K_{min}) – 2	8N + 4(K_{max}-K_{min})
Weighted (divisible) Vote = T shares	2TN [+3N]	4TN [+5N]	10N + 4 *	5N + 2 *	(4T-2)(N+1)	4T(N+1)
Rank K-out-of-N	2N [+2KN +3N]	4N [+4KN +5N]	4N + 2 **	2N + 3 **	6(N+1)K + 2K	8(N+1)K + 4K

* Does not support a limit per candidate.

** Ranks all candidates and limits the homomorphic tally to the Borba method.

Related work – continuation I

- Groth 2005
  - Complexity grows exponentially with the number of candidates and the number of votes allowed in the homomorphic tally.

  - Large number of candidates => large exponents size
    - Exponent size ~ $\log_2$ (#votes) * # candidates

  - Requires a crypto system with an easy decryption of

    E(m1+m2, r1+r2) = E(m1, r1)E(m2, r2), e.g. Paillier.
    - Size of decryption table for 256 bits EC-ElGamal

      10 candidates, 100 votes -> more than 25TB!!!

In practice, does not work for complex elections.

Related work – continuation II

- Helios
  - Direct mix-net tallying is expensive because it involves one ciphertext per each possible option (candidate).
  - The number of ciphertexts can be reduced by using more expensive proofs.
  - No mix-net solution for ranked candidates.
  - Larger proofs.

A NEW WAY TO VERIFY AN ENCRYPTION OF A COMPLEX VOTE

Key idea

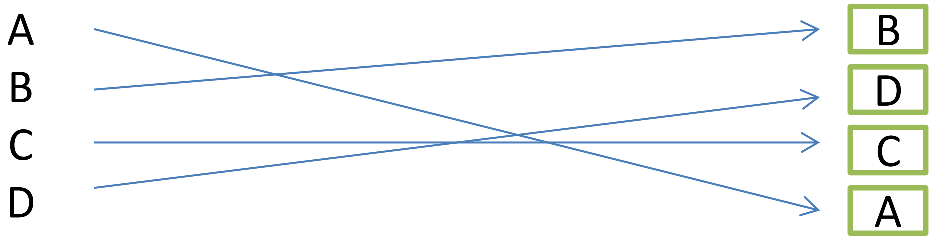
1. Create a verifiable shuffle of a set of candidate identifiers.
    - Bayer and Groth 2012

2. Create the vote encryption directly from the shuffle output.

3. Add ZKPK to check ballot structure constrains.

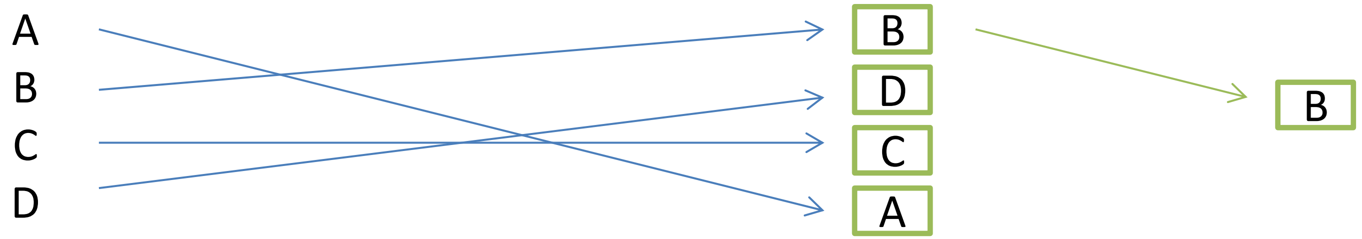
Approval voting

Candidate Identifiers	Verifiable Shuffle	Shuffle of the encrypted candidate identifiers	Vote


A
B
C
D

B
D
C
A

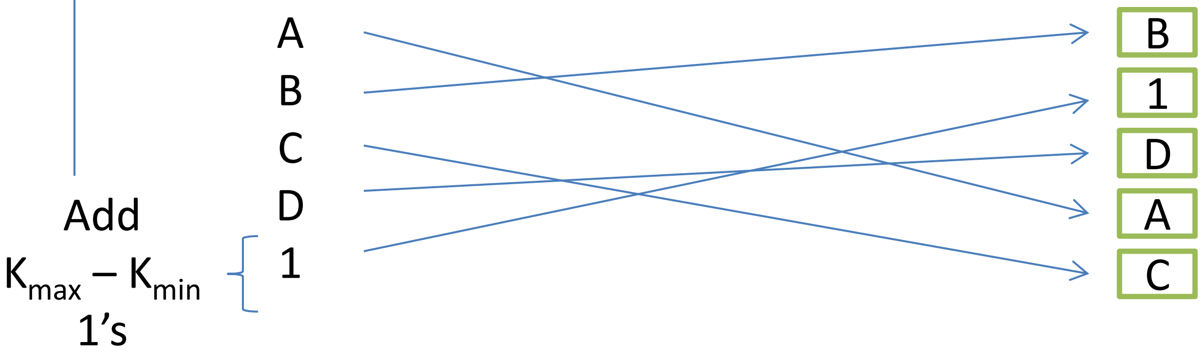
Approval voting



Candidate Identifiers	Verifiable Shuffle	Shuffle of the encrypted candidate identifiers	Vote
A B C D		B D C A	B

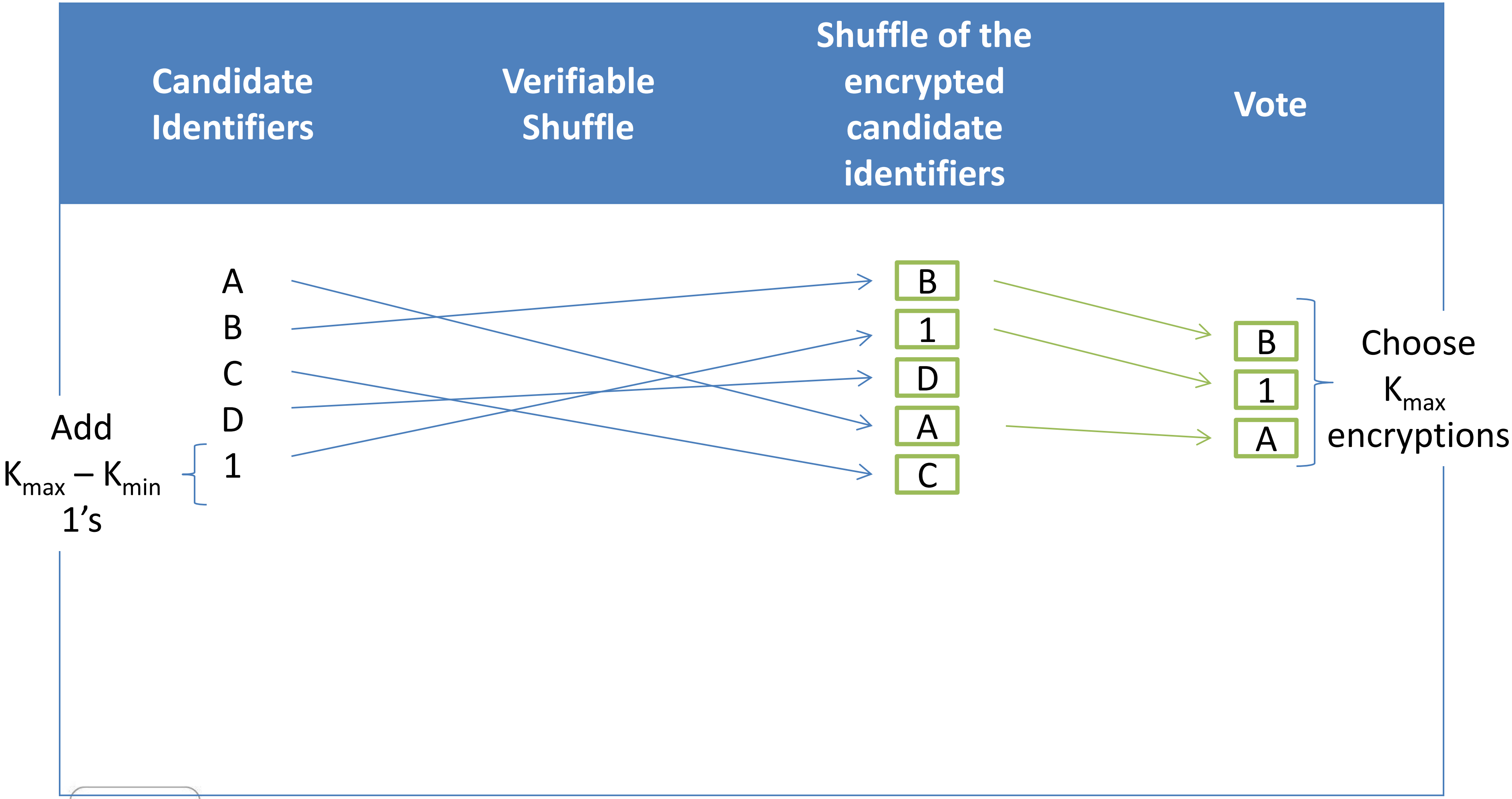
Approval voting [K_{min} , K_{max}]
Example : [2-3]

Candidate Identifiers	Verifiable Shuffle	Shuffle of the encrypted candidate identifiers	Vote



Add $K_{max} – K_{min}$ 1's

Approval voting [K_{min} , K_{max}]
Example : [2-3]



Candidate Identifiers

Verifiable Shuffle

Shuffle of the encrypted candidate identifiers

Vote

A
B
C
D
1

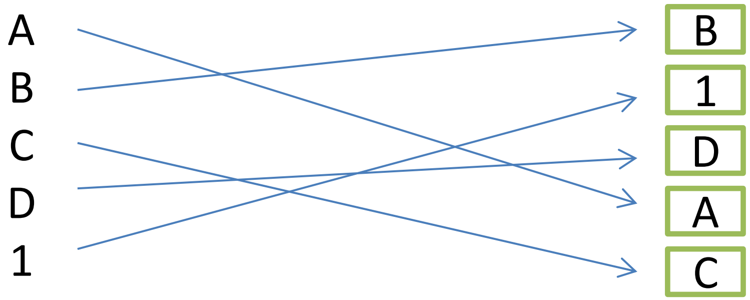
Add $K_{max} - K_{min}$ 1's

B
1
D
A
C

B
1
A

Choose K_{max} encryptions

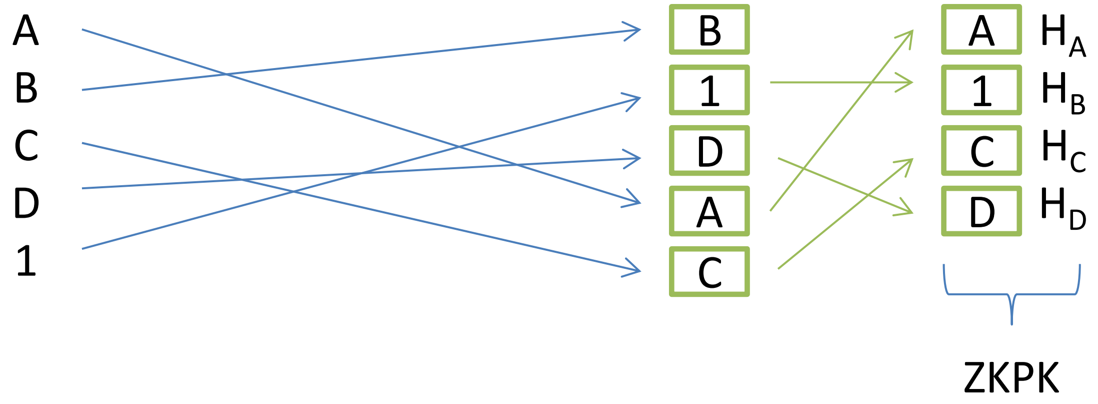
Approval voting with homomorphic tally



Candidate Identifiers	Verifiable Shuffle	Shuffle of the encrypted candidate identifiers	Homomorphic counters	Vote
A B C D 1		B 1 D A C	H_A H_B H_C H_D	

Approval voting with homomorphic tally

Candidate Identifiers	Verifiable Shuffle	Shuffle of the encrypted candidate identifiers	Homomorphic counters	Vote



Approval voting with homomorphic tally

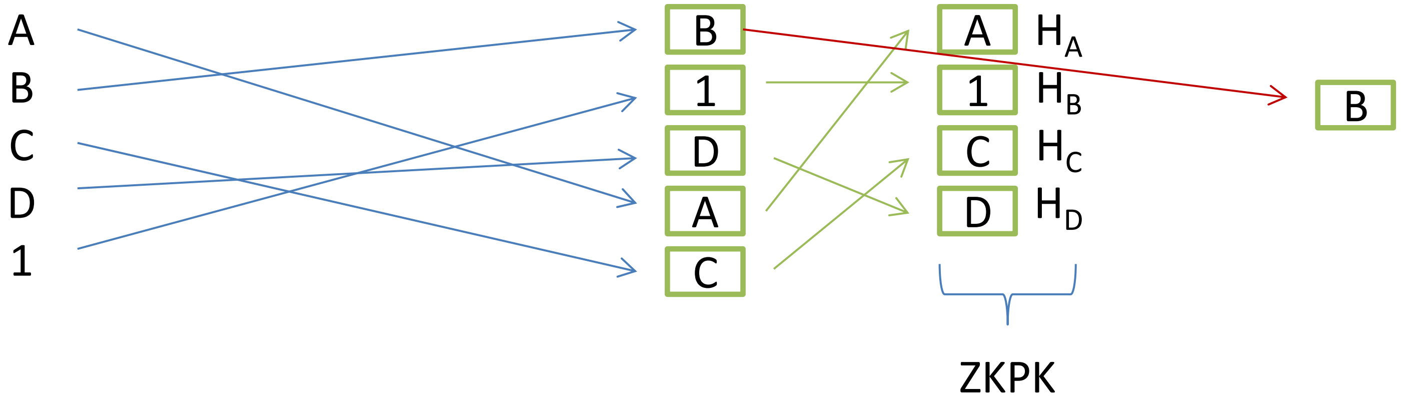
Candidate Identifiers

Verifiable Shuffle

Shuffle of the encrypted candidate identifiers

Homomorphic counters

Vote



A
B
C
D
1

B
1
D
A
C

A H_A
1 H_B
C H_C
D H_D

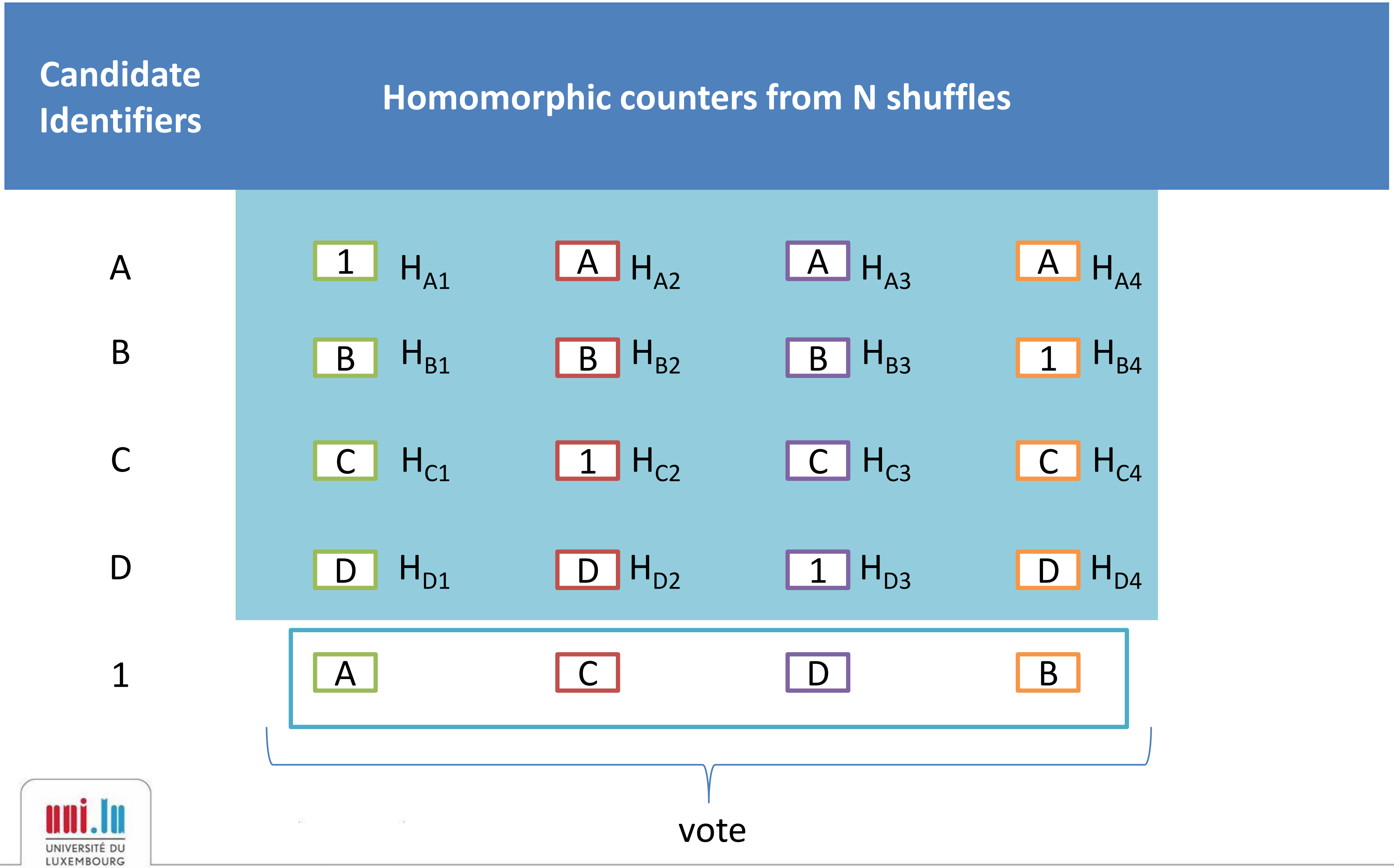
ZKPK

B

Ranked voting with homomorphic tally

Candidate Identifiers	Homomorphic counters from N shuffles			
A	$\boxed{1}$ H_{A1}	$\boxed{A}$ H_{A2}	$\boxed{A}$ H_{A3}	$\boxed{A}$ H_{A4}
B	$\boxed{B}$ H_{B1}	$\boxed{B}$ H_{B2}	$\boxed{B}$ H_{B3}	$\boxed{1}$ H_{B4}
C	$\boxed{C}$ H_{C1}	$\boxed{1}$ H_{C2}	$\boxed{C}$ H_{C3}	$\boxed{C}$ H_{C4}
D	$\boxed{D}$ H_{D1}	$\boxed{D}$ H_{D2}	$\boxed{1}$ H_{D3}	$\boxed{D}$ H_{D4}
1				

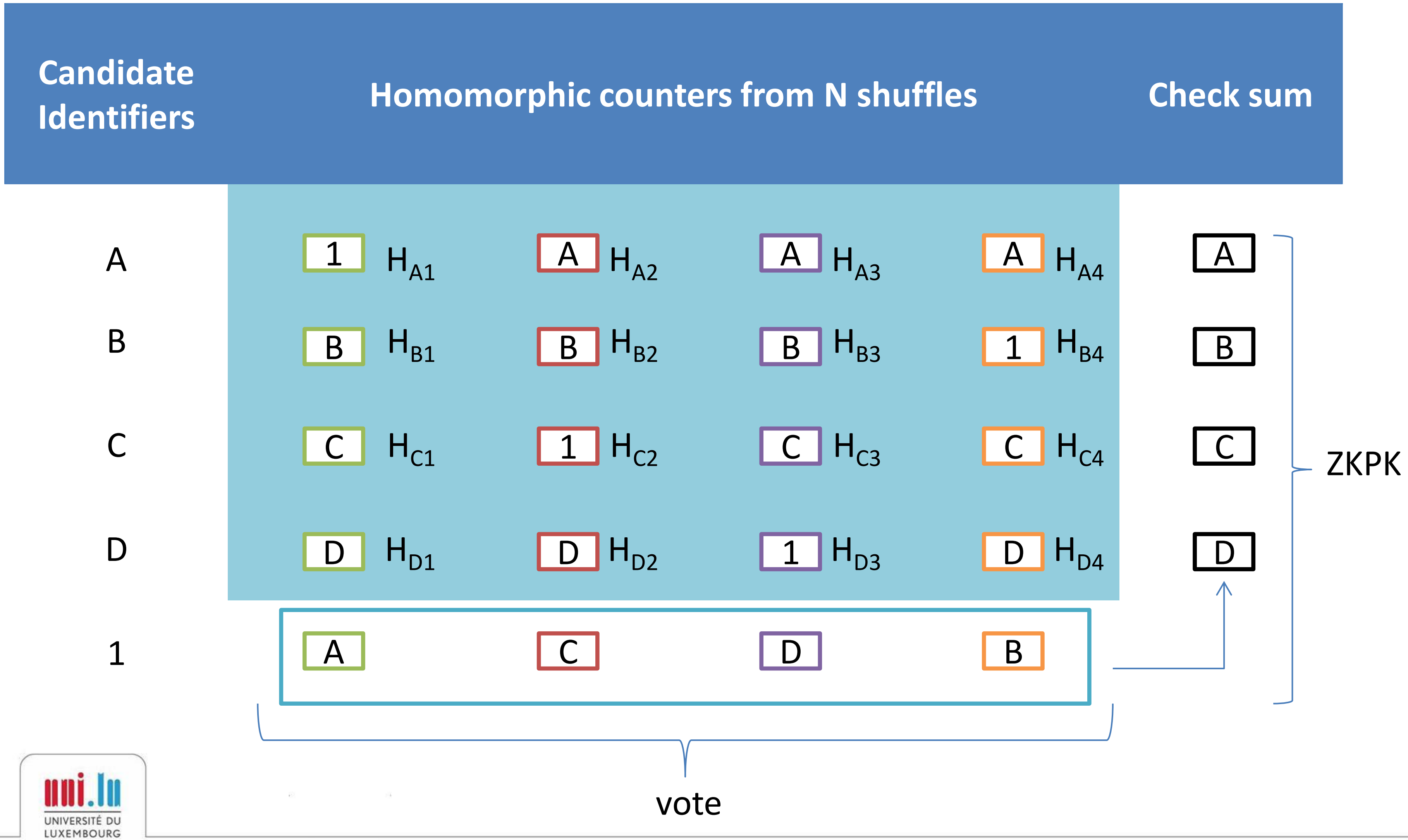
Ranked voting with homomorphic tally



Candidate Identifiers	Homomorphic counters from N shuffles			
A	1 H_{A1}	A H_{A2}	A H_{A3}	A H_{A4}
B	B H_{B1}	B H_{B2}	B H_{B3}	1 H_{B4}
C	C H_{C1}	1 H_{C2}	C H_{C3}	C H_{C4}
D	D H_{D1}	D H_{D2}	1 H_{D3}	D H_{D4}
1	A	C	D	B

vote

Ranked voting with homomorphic tally



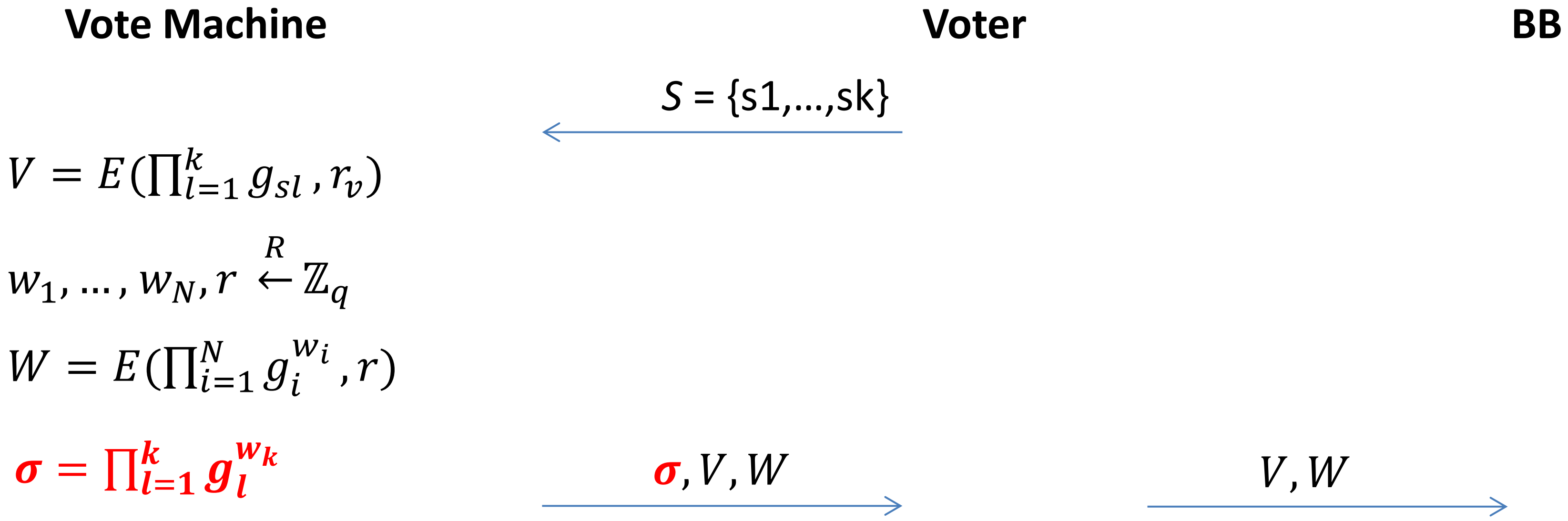
Candidate Identifiers	Homomorphic counters from N shuffles				Check sum
A	1 H_{A1}	A H_{A2}	A H_{A3}	A H_{A4}	A
B	B H_{B1}	B H_{B2}	B H_{B3}	1 H_{B4}	B
C	C H_{C1}	1 H_{C2}	C H_{C3}	C H_{C4}	C
D	D H_{D1}	D H_{D2}	1 H_{D3}	D H_{D4}	D
1	A	C	D	B	

A SINGLE VERIFICATION CODE FOR A COMPLEX BALLOT

Preliminaries

- Consider:
  - The usual ElGamal setup.
  - One independent generator (g_i) for every choice/candidate i.
  - Let $V = E(\prod_{l=1}^{k} g_{jl}, r_v)$ be the homomorphic product of the k ciphertexts that compose the vote.

The voter verification protocol (v1)



Vote Machine **Voter** **BB**

$S = \{s1,…,sk\}$

$$V = E(\prod_{l=1}^{k} g_{sl}, r_v)$$

$$w_1, \dots, w_N, r \overset{R}{\leftarrow} \mathbb{Z}_q$$

$$W = E(\prod_{i=1}^{N} g_i^{w_i}, r)$$

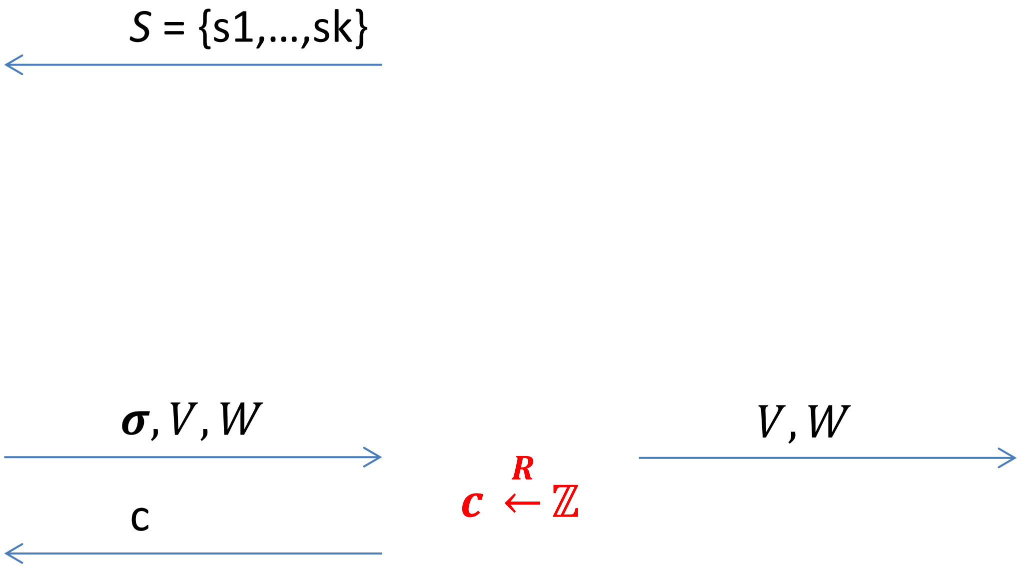
$$\boldsymbol{\sigma = \prod_{l=1}^{k} g_l^{w_k}}$$

$\boldsymbol{\sigma}, V, W$

V, W

The voter verification protocol (v1)

Vote Machine **Voter** **BB**



S = {s1,…,sk}

$$V = E(\prod_{l=1}^{k} g_{sl}, r_v)$$

$$w_1, \dots, w_N, r \overset{R}{\leftarrow} \mathbb{Z}_q$$

$$W = E(\prod_{i=1}^{N} g_i^{w_i}, r)$$

$$\boldsymbol{\sigma = \prod_{l=1}^{k} g_l^{w_k}}$$

$\boldsymbol{\sigma}, V, W$

c

$\boldsymbol{c \overset{R}{\leftarrow} \mathbb{Z}}$

V, W

The voter verification protocol (v1)

Vote Machine **Voter** **BB**

$\xleftarrow{S = \{s1,\dots,sk\}}$

$V = E(\prod_{l=1}^{k} g_{sl}, r_v)$

$w_1, \dots, w_N, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\prod_{i=1}^{N} g_i^{w_i}, r)$

$\boldsymbol{\sigma = \prod_{l=1}^{k} g_l^{w_k}}$ $\xrightarrow{\boldsymbol{\sigma}, V, W}$ $\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$ $\xrightarrow{V, W}$

$\xleftarrow{c}$

$\forall s_i \in S: r_{si} = w_{si} + c$

$\forall l \notin S: r_l = w_l$

$$P_r = ZKPK[W \otimes V^c = E(\prod_{i=1}^{N} g_i^{r_i}, r')]$$

$\xrightarrow{r_1, \dots, r_n, P_r}$ $\xrightarrow{c, r_1, \dots, r_n, P_r}$

$\boldsymbol{P_r}$!?

$$\boldsymbol{\sigma} \mathrel{?=} \prod\nolimits_{si \in S} g_{si}^{r_{si} - c}$$

The voter verification protocol (v1)

Vote Machine **Voter** **BB**

$$\xleftarrow{S = \{s1,\ldots,sk\}}$$

$$V = E(\textstyle\prod_{l=1}^{k} g_{sl}, r_v)$$

$$w_1, \ldots, w_N, r \overset{R}{\leftarrow} \mathbb{Z}_q$$

$$W = E(\textstyle\prod_{i=1}^{N} g_i^{w_i}, r)$$

$$\boldsymbol{\sigma} = \textstyle\prod_{l=1}^{k} \boldsymbol{g_l^{w_k}} \quad \xrightarrow{\boldsymbol{\sigma}, V, W} \quad \boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z} \quad \xrightarrow{V, W}$$

$$\xleftarrow{c}$$

$$\forall s_i \in S: r_{si} = w_{si} + c$$

$$\forall l \notin S: r_l = w_l$$

$$P_r = ZKPK[W \otimes V^c = E(\prod_{i=1}^{N} g_i^{r_i}, r')]$$

$$\xrightarrow{r_1, \ldots, r_n, P_r} \qquad \xrightarrow{c, r_1, \ldots, r_n, P_r}$$

$$\boldsymbol{P_r}!?$$

$$\boldsymbol{\sigma}\ ?= \prod\nolimits_{si \in S} \boldsymbol{g_{si}^{r_{si}-c}}$$

The voter verification protocol (v1)

Vote Machine | **Voter** | **BB**

$\xleftarrow{S = \{s1,\ldots,sk\}}$

$V = E(\prod_{l=1}^{k} g_{sl}, r_v)$

$w_1, \ldots, w_N, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\prod_{i=1}^{N} g_i^{w_i}, r)$

$\boldsymbol{\sigma = \prod_{l=1}^{k} g_l^{w_k}}$

$\xrightarrow{\boldsymbol{\sigma}, V, W}$ (Voter) $\xrightarrow{V, W}$ (BB)

$\boldsymbol{c} \overset{R}{\leftarrow} \mathbb{Z}$

$\xleftarrow{\text{c}}$

$\forall s_i \in S: r_{si} = w_{si} + c$

$\forall l \notin S: r_l = w_l$

$P_r = ZKPK[W \otimes V^c = E(\prod_{i=1}^{N} g_i^{r_i}, r')]$

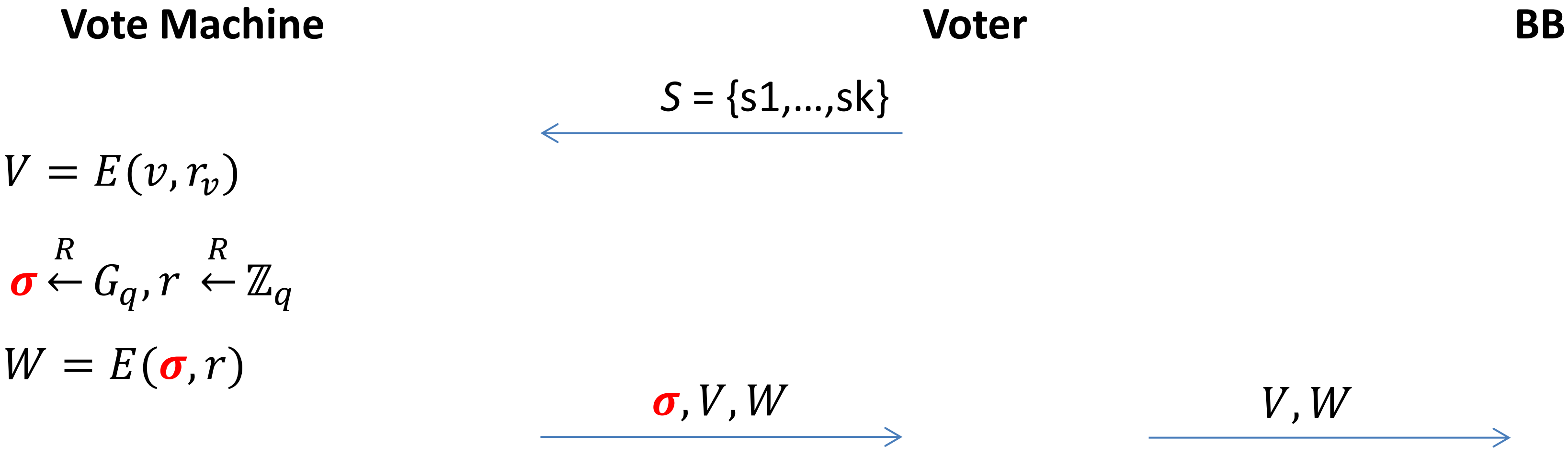
$\xrightarrow{r_1, \ldots, r_n, P_r}$ (Voter) $\xrightarrow{c, r_1, \ldots, r_n, P_r}$ (BB)

$\boldsymbol{\sigma ?= \prod_{si \in S} g_{si}^{r_{si}-c}}$

$\boldsymbol{P_r}!?$

UNIVERSAL A VERIFICATION CODE

The voter verification protocol (v2)



Vote Machine

Voter

BB

S = {s1,…,sk}

$$V = E(v, r_v)$$

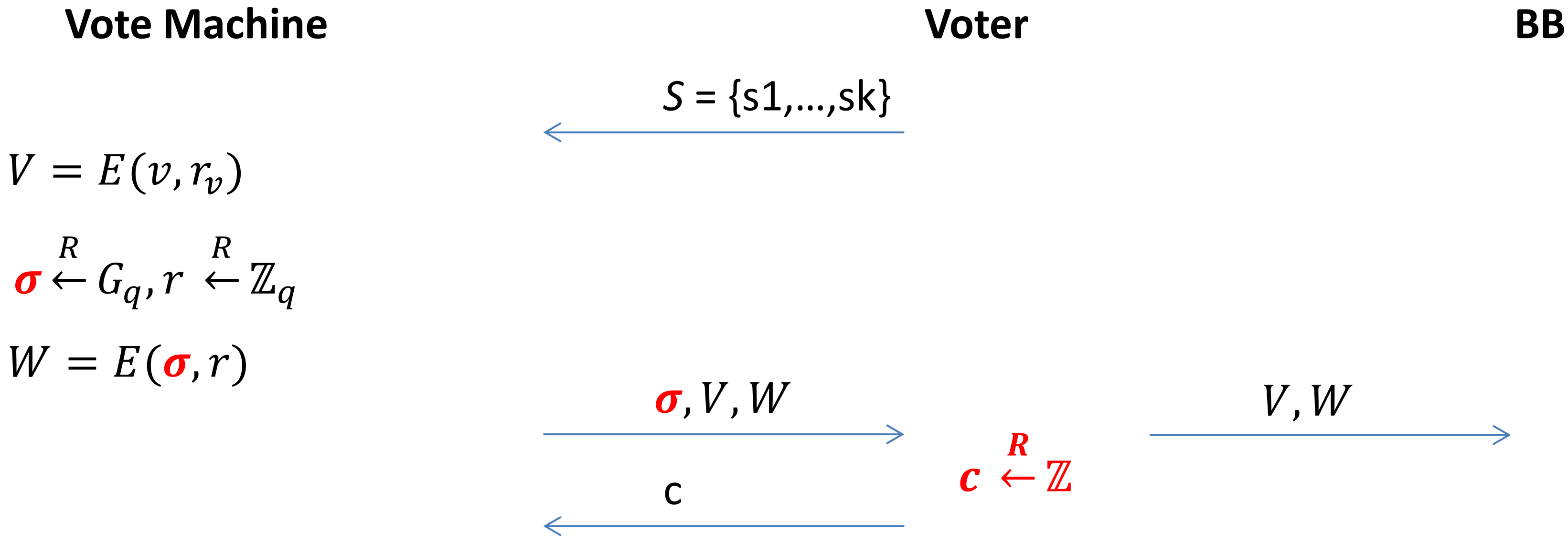
$$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$$

$$W = E(\boldsymbol{\sigma}, r)$$

$\boldsymbol{\sigma}, V, W$

V, W

The voter verification protocol (v2)



Vote Machine

Voter

BB

S = {s1,…,sk}

$V = E(v, r_v)$

$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\boldsymbol{\sigma}, r)$

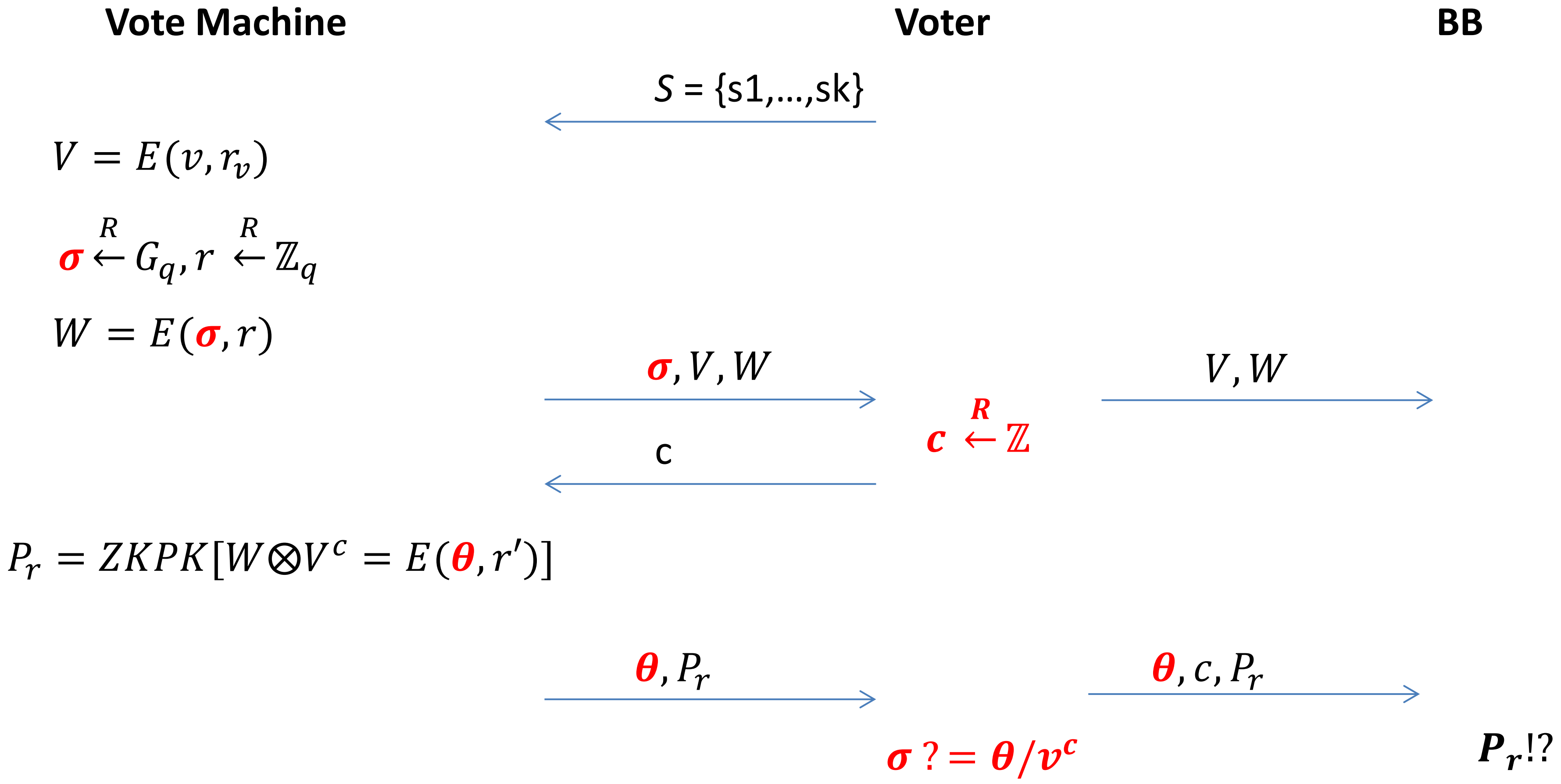
$\boldsymbol{\sigma}, V, W$

V, W

$\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$

c

The voter verification protocol (v2)



Vote Machine **Voter** **BB**

$\xleftarrow{S = \{s1,\ldots,sk\}}$

$V = E(v, r_v)$

$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\boldsymbol{\sigma}, r)$

$\xrightarrow{\boldsymbol{\sigma}, V, W}$ $\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$ $\xrightarrow{V, W}$

$\xleftarrow{c}$

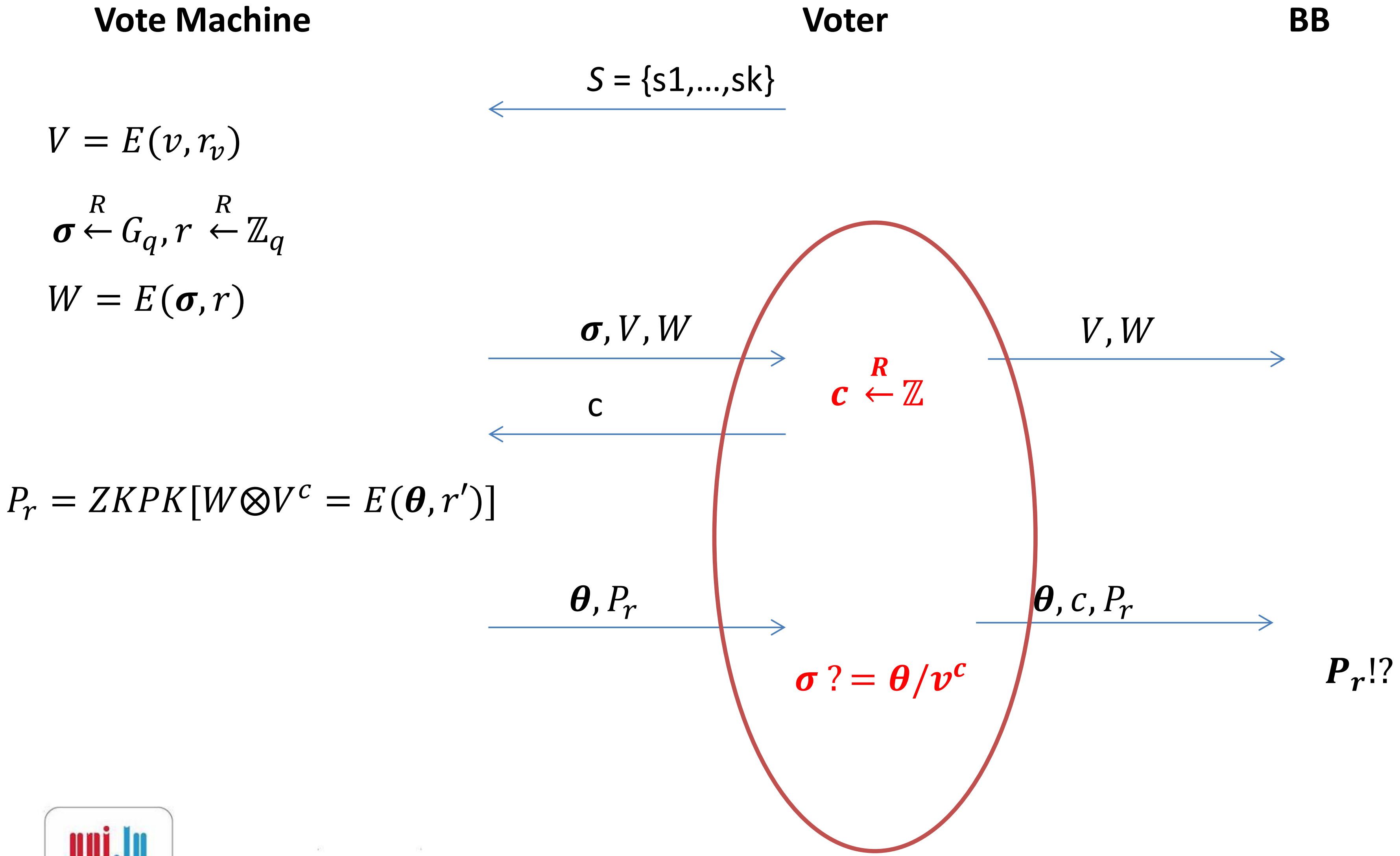
$P_r = ZKPK[W \otimes V^c = E(\boldsymbol{\theta}, r')]$

$\xrightarrow{\boldsymbol{\theta}, P_r}$ $\xrightarrow{\boldsymbol{\theta}, c, P_r}$

$\boldsymbol{\sigma} \,?\!= \boldsymbol{\theta} / \boldsymbol{v}^{\boldsymbol{c}}$

$\boldsymbol{P_r}!?$

The voter verification protocol (v2)



Vote Machine

Voter

BB

S = {s1,…,sk}

$V = E(v, r_v)$

$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\boldsymbol{\sigma}, r)$

$\boldsymbol{\sigma}, V, W$

V, W

$\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$

c

$P_r = ZKPK[W \otimes V^c = E(\boldsymbol{\theta}, r')]$

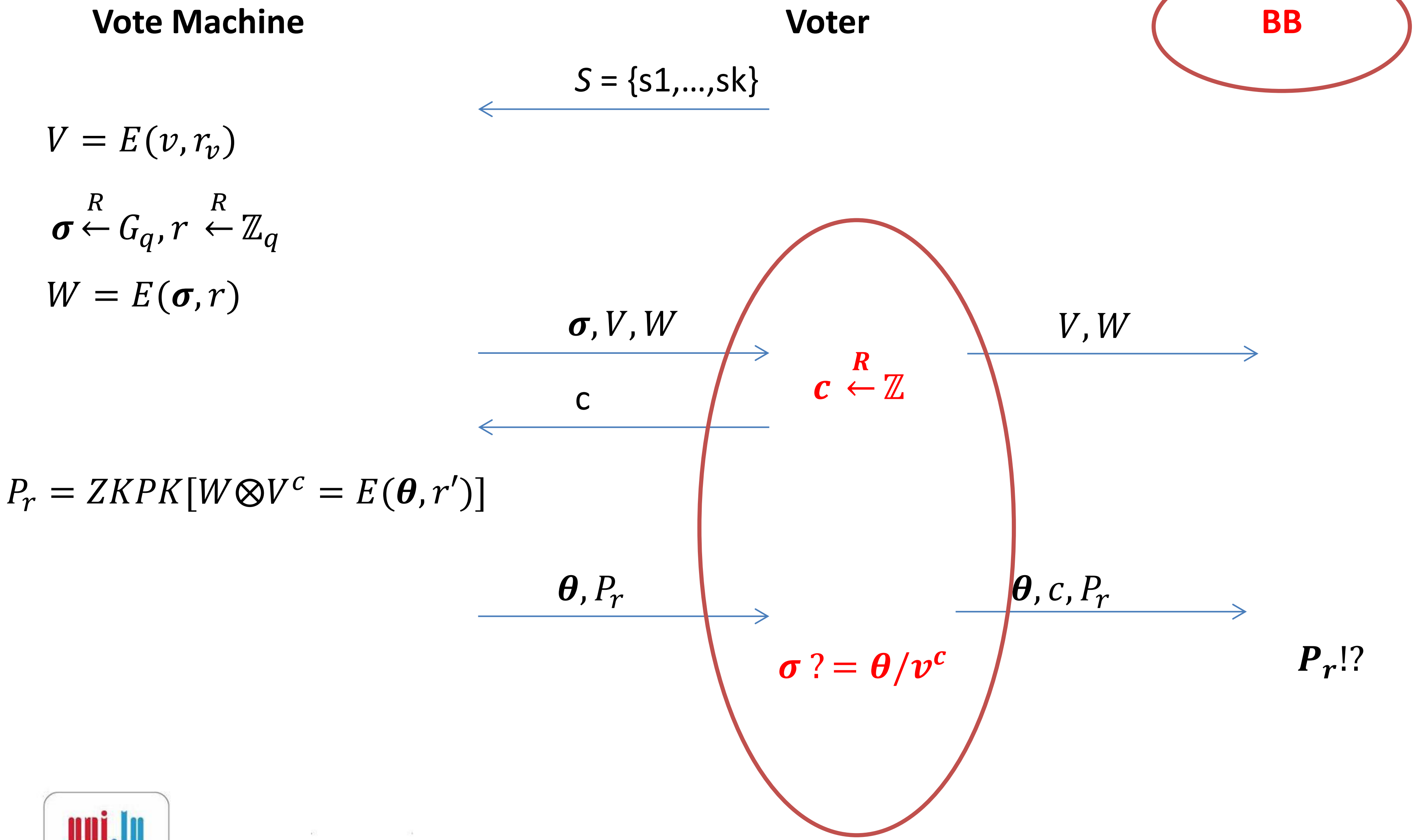
$\boldsymbol{\theta}, P_r$

$\boldsymbol{\theta}, c, P_r$

$\boldsymbol{\sigma}\ ? = \boldsymbol{\theta} / \boldsymbol{v}^{\boldsymbol{c}}$

$\boldsymbol{P_r}$!?

The voter verification protocol (v2)



Vote Machine

Voter

BB

$S = \{s1,\ldots,sk\}$

$V = E(v, r_v)$

$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\boldsymbol{\sigma}, r)$

$\boldsymbol{\sigma}, V, W$

V, W

$\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$

c

$P_r = ZKPK[W \otimes V^c = E(\boldsymbol{\theta}, r')]$

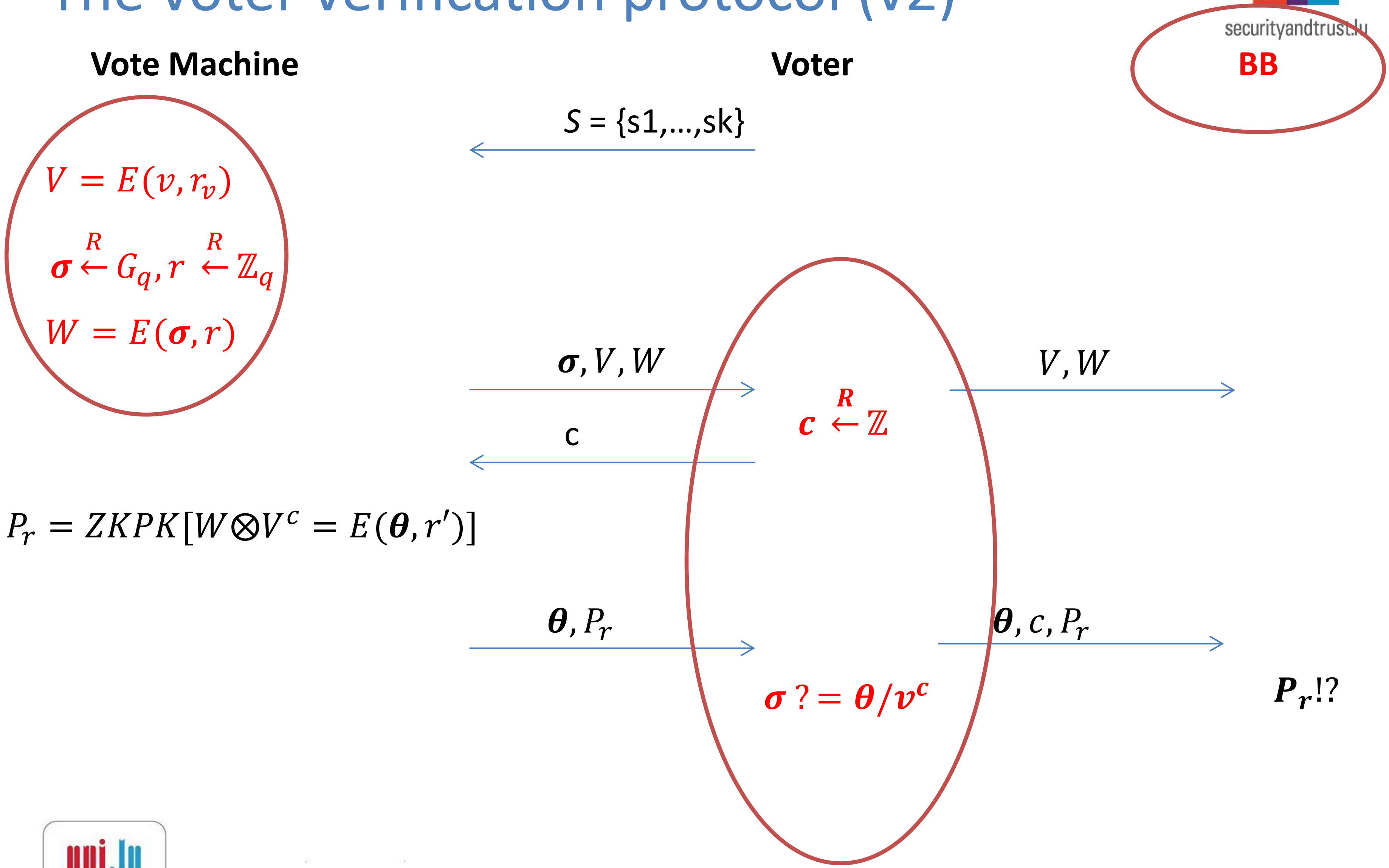
$\boldsymbol{\theta}, P_r$

$\boldsymbol{\theta}, c, P_r$

$\boldsymbol{\sigma} \mathrel{?=} \boldsymbol{\theta} / \boldsymbol{v}^{\boldsymbol{c}}$

$\boldsymbol{P_r}$!?

The voter verification protocol (v2)



Vote Machine

Voter

BB

S = {s1,…,sk}

$V = E(v, r_v)$

$\boldsymbol{\sigma} \overset{R}{\leftarrow} G_q, r \overset{R}{\leftarrow} \mathbb{Z}_q$

$W = E(\boldsymbol{\sigma}, r)$

$\boldsymbol{\sigma}, V, W$

V, W

$\boldsymbol{c} \overset{\boldsymbol{R}}{\leftarrow} \mathbb{Z}$

c

$P_r = ZKPK[W \otimes V^c = E(\boldsymbol{\theta}, r')]$

$\boldsymbol{\theta}, P_r$

$\boldsymbol{\theta}, c, P_r$

$\boldsymbol{\sigma} ?= \boldsymbol{\theta}/\boldsymbol{v}^{\boldsymbol{c}}$

$\boldsymbol{P_r}$!?

Thank you!

Questions?