Verifiable Homomorphic Tallying for the Schulze Vote Counting Scheme

Mukesh Tiwari[1] Dirk Pattinson[1] Thomas Haines [2]

[1]Research School of Computer Science, Australian National University, Australia

[2]Department of Mathematical Sciences, Norwegian University of Science and Technology, Norway

July 10, 2019

Outline

- Motivation (Privacy and Verifiability)
- Why Coq ?
- Schulze Method
- Homomorphic Schulze
- Experimental Result

Motivation (Verifiability)

"Those who cast the vote decide nothing. Those who count the vote decide everything." Joseph Stalin

Motivation (Privacy)

"The villages that cast 80 per cent votes for Bharatiya Janata Party will be put in category A, those that cast 60 per cent votes will be in category B and so on and so forth. Villages in A category will get priority in development and then will come the turn of other categories. It is up to you whether you make it to A, B, C or D. No one should fall in D category!" Maneka Gandhi

Verifiable Voting Scheme

- Cast as Intended
- Recorded as Cast
- Tallied as Recorded

Verifiable Voting Scheme (Bulletin Board)

Dirk : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Mina : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Caitlin : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Raj : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Ranald : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Thomas : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
==================================

Verifiable Voting Scheme (Bulletin Board)

Dirk : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Mina : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Caitlin : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Raj : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Ranald : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
Thomas : [Maneka : 1, B : 2, C : 3, D : 4, E : 5]
==

Privacy Preserving Voting Scheme

The winner is

Privacy Preserving Voting Scheme

The winner is

Why Coq and why not JavaScript ?

Schulze Method (Plain Text Ballot)

- Consider an election with a set of m candidates $C = \{c1, \ldots, cm\}$, and a multi-set of n votes $P = \{b1, \ldots, bn\}$. A (plaintext) ballot is represented as function $b : C \rightarrow \mathbb{N}$ that assigns natural number (the preference) to each candidate.

Rank all candidates
in order of preference

[4] Lando Calrissian

[3] Boba Fett

[1] Mace Windu

[2] Poe Dameron

[2] Maz Kanata

Schulze Method (Margin Matrix)

- Given two candidates $c, d \in C$, the *margin* of c over d is the number of voters that prefer c over d, minus the number of voters that prefer d over c.

$$m(c, d) = \sharp\{b \in P \mid c >_b d\} - \sharp\{b \in P \mid d >_b c\}$$

where $\sharp$ denotes cardinality and $>_b$ is the ordering given by the ballot b.

Schulze Method (Generalized Margin Matrix)

- A directed *path* from candidate c to candidate d is a sequence $p \equiv c_0, \ldots, c_{n+1}$ of candidates with $c_0 = c$ and $c_{n+1} = d$ $(n \geq 0)$, and the *strength*, st, of path, p, is the minimum margin of adjacent nodes, i.e.

$$st(c_0, \ldots, c_{n+1}) = \min\{m(c_i, c_{i+1}) \mid 0 \leq i \leq n\}.$$

Schulze Method (Generalized Margin Matrix)

- A directed *path* from candidate c to candidate d is a sequence $p \equiv c_0, \ldots, c_{n+1}$ of candidates with $c_0 = c$ and $c_{n+1} = d$ $(n \geq 0)$, and the *strength*, st, of path, p, is the minimum margin of adjacent nodes, i.e.

$$st(c_0, \ldots, c_{n+1}) = \min\{m(c_i, c_{i+1}) \mid 0 \leq i \leq n\}.$$

- For candidates c and d, let $M(c, d)$ denote the maximum strength, or generalized margin of a path from c to d i.e.

$$M(c, d) = \max\{st(p) : \text{p is path from c to d in G}\}$$

Schulze Method (Generalized Margin Matrix)

- A directed *path* from candidate c to candidate d is a sequence $p \equiv c_0, \ldots, c_{n+1}$ of candidates with $c_0 = c$ and $c_{n+1} = d$ $(n \geq 0)$, and the *strength*, st, of path, p, is the minimum margin of adjacent nodes, i.e.

$$st(c_0, \ldots, c_{n+1}) = \min\{m(c_i, c_{i+1}) \mid 0 \leq i \leq n\}.$$

- For candidates c and d, let $M(c, d)$ denote the maximum strength, or generalized margin of a path from c to d i.e.

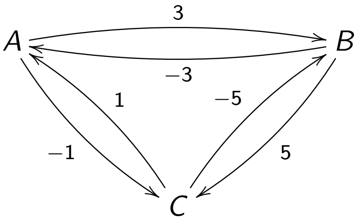
$$M(c, d) = \max\{st(p) : \text{p is path from c to d in G}\}$$

- The winning set (always non empty) is defined as

$$W = \{c \in C : \forall d \in C \setminus \{c\}, M(c, d) \geq M(d, c)\}$$

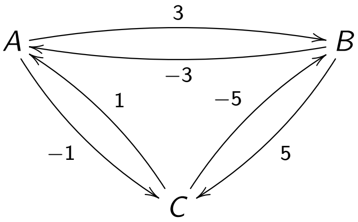
Example of Schulze method

- Margin matrix

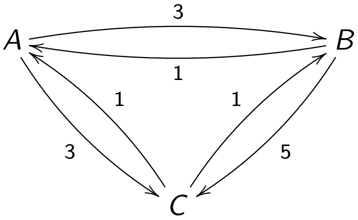


Example of Schulze method

- Margin matrix

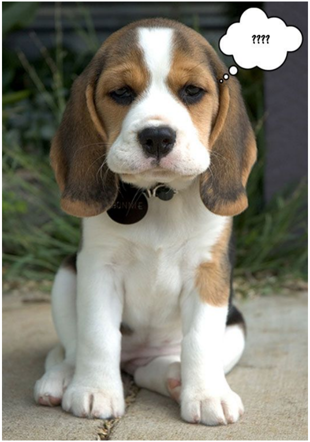


- Computing generalized margin



A Question to Ponder

Can we guess the ballots from Margin matrix ?



Homomorphic Schulze (Pillars)

- Is it possible to convince the voter that we (Electoral Authority) have counted all the ballots honestly ?

Homomorphic Schulze (Pillars)

- Is it possible to convince the voter that we (Electoral Authority) have counted all the ballots honestly ?
- Homomorphic Encryption
- Zero Knowledge Proof

Ballot Representation for Homomorphic Schulze

- Plaintext ballot

$$\begin{array}{cc} & \begin{array}{ccc} \mathit{Dirk} & \mathit{Mina} & \mathit{Caity} \end{array} \\ \begin{array}{c} \mathit{Dirk} \\ \mathit{Mina} \\ \mathit{Caity} \end{array} & \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix} \end{array}$$

Ballot Representation for Homomorphic Schulze

- Plaintext ballot

$$\begin{array}{c c} & \begin{array}{ccc} \mathit{Dirk} & \mathit{Mina} & \mathit{Caity} \end{array} \\ \begin{array}{c} \mathit{Dirk} \\ \mathit{Mina} \\ \mathit{Caity} \end{array} & \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix} \end{array}$$

- Ciphertext ballot

$$\begin{array}{c c} & \begin{array}{ccc} \mathit{Dirk} & \mathit{Mina} & \mathit{Caity} \end{array} \\ \begin{array}{c} \mathit{Dirk} \\ \mathit{Mina} \\ \mathit{Caity} \end{array} & \begin{pmatrix} (42..,15..) & (63..,54..) & (89..,67..) \\ (16..,43..) & (12..,46..) & (71..,11..) \\ (96..,67..) & (54..,43..) & (39..,28..) \end{pmatrix} \end{array}$$

Homomorphic Margin Computation

- Margin from plaintext ballots

$$m(c, d) = \sum_{b \in B} b(x, y)$$

Homomorphic Margin Computation

- Margin from plaintext ballots

$$m(c, d) = \sum_{b \in B} b(x, y)$$

- Encrypted margin from ciphertext ballots

$$encm(c, d) = \bigoplus_{encb \in EncB} encb(x, y)$$

Homomorphic Margin Computation

- Margin from plaintext ballots

$$m(c, d) = \sum_{b \in B} b(x, y)$$

- Encrypted margin from ciphertext ballots

$$encm(c, d) = \bigoplus_{encb \in EncB} encb(x, y)$$

- In additive Elgamal encryption, $E(m) = (g^r, g^m * h^r)$. We can easily verify that $E(m_1) * E(m_2) = E(m_1 + m_2)$.

Problems with Matrix Representation

- Voter can inflate your ballot

$$
\begin{array}{cc}
 & \begin{array}{ccc} \textit{Dirk} & \textit{Mina} & \textit{Caity} \end{array} \\
\begin{array}{c} \textit{Dirk} \\ \textit{Mina} \\ \textit{Caity} \end{array} &
\begin{pmatrix} 0 & 10 & 10 \\ -10 & 0 & 10 \\ -10 & -10 & 0 \end{pmatrix}
\end{array}
$$

Problems with Matrix Representation

- Voter can inflate your ballot

$$\begin{array}{c} \\ Dirk \\ Mina \\ Caity \end{array} \begin{array}{c} \begin{array}{ccc} Dirk & Mina & Caity \end{array} \\ \begin{pmatrix} 0 & 10 & 10 \\ -10 & 0 & 10 \\ -10 & -10 & 0 \end{pmatrix} \end{array}$$

- Voter can construct cyclic ballot

$$\begin{array}{c} \\ Dirk \\ Mina \\ Caity \end{array} \begin{array}{c} \begin{array}{ccc} Dirk & Mina & Caity \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix} \end{array}$$

Solution (Protocol Design)

- We take a ballot (u)

$$\begin{array}{cc} & \begin{array}{ccc} \mathit{Dirk} & \mathit{Mina} & \mathit{Caity} \end{array} \\ \begin{array}{c} \mathit{Dirk} \\ \mathit{Mina} \\ \mathit{Caity} \end{array} & \begin{pmatrix} (42.., 15..) & (63.., 54..) & (89.., 67..) \\ (16.., 43..) & (12.., 46..) & (71.., 11..) \\ (96.., 67..) & (54.., 43..) & (39.., 28..) \end{pmatrix} \end{array}$$

Solution (Protocol Design)

- We take a ballot (u)

$$\begin{array}{c c} & \begin{array}{ccc} \textit{Dirk} & \textit{Mina} & \textit{Caity} \end{array} \\ \begin{array}{c} \textit{Dirk} \\ \textit{Mina} \\ \textit{Caity} \end{array} & \begin{pmatrix} (42..,15..) & (63..,54..) & (89..,67..) \\ (16..,43..) & (12..,46..) & (71..,11..) \\ (96..,67..) & (54..,43..) & (39..,28..) \end{pmatrix} \end{array}$$

- We generate a secret random permutation, σ, publish its commitment, and zero knowledge proof.

Solution (Protocol Design)

- We take a ballot (u)

$$\begin{array}{cc} & \begin{array}{ccc} \textit{Dirk} & \textit{Mina} & \textit{Caity} \end{array} \\ \begin{array}{c} \textit{Dirk} \\ \textit{Mina} \\ \textit{Caity} \end{array} & \begin{pmatrix} (42.., 15..) & (63.., 54..) & (89.., 67..) \\ (16.., 43..) & (12.., 46..) & (71.., 11..) \\ (96.., 67..) & (54.., 43..) & (39.., 28..) \end{pmatrix} \end{array}$$

- We generate a secret random permutation, σ, publish its commitment, and zero knowledge proof.
- We permute each row of ballot u by σ which produces row-shuffled ballot (v) and zero knowledge proof

$$\begin{array}{cc} & \begin{array}{ccc} \textit{Dirk} & \textit{Mina} & \textit{Caity} \end{array} \\ \begin{array}{c} \textit{Dirk} \\ \textit{Mina} \\ \textit{Caity} \end{array} & \begin{pmatrix} (36.., 97..) & (81.., 51..) & (12.., 98..) \\ (31.., 23..) & (78.., 67..) & (19.., 41..) \\ (76.., 44..) & (31.., 61..) & (43.., 22..) \end{pmatrix} \end{array}$$

Solution (Protocol Design)

- We permute each column of ballot v by σ (w) in similar fashion

$$
\begin{array}{cc}
 & \begin{array}{ccc} \textit{Dirk} & \textit{Mina} & \textit{Caity} \end{array} \\
\begin{array}{c} \textit{Dirk} \\ \textit{Mina} \\ \textit{Caity} \end{array} &
\begin{pmatrix}
(31.., 44..) & (73.., 35..) & (43.., 65..) \\
(82.., 36..) & (56.., 82..) & (27.., 23..) \\
(67.., 38..) & (15.., 91..) & (89.., 98..)
\end{pmatrix}
\end{array}
$$

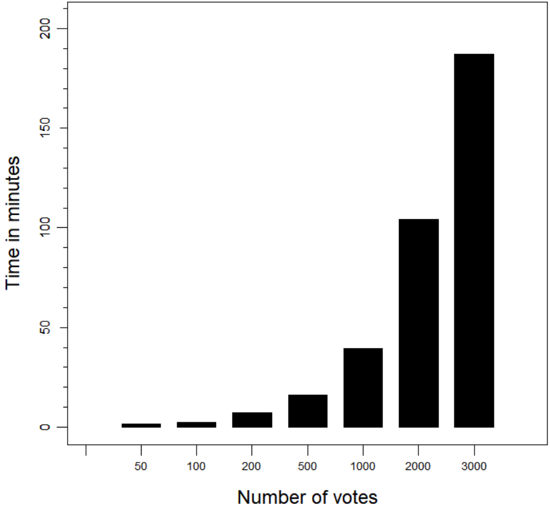
- We decrypt the encrypted ballot w into plain text ballot b with zero knowledge proof that b is indeed honest decryption of w.

Correctness Proof

If there one to one correspondence between plaintext ballots and encrypted ballots, then computing winners via plaintext ballot is same as encrypted ballot

```
forall (bs : list (ballot cand))
    (ebs : list (eballot cand))
    (w : cand -> bool),
    (* some details omitted *)
    Count cand cand_all bs (winners cand w) <->
    ECount cand cand_all ebs (ewinners cand w)
```

Experimental Result



Thank You!