Minimum Disclosure Counting for the Alternative Vote

Roland Wen and Richard Buckland

School of Computer Science and Engineering
The University of New South Wales
Sydney, Australia
{rolandw,richardb}@cse.unsw.edu.au

VOTE-ID 2009

Outline

The Alternative Vote

- Preferential electoral system
  - Voters express preferences for all candidates
- Alternative vote
  - Elect single candidate
  - Winner must obtain majority ($> 50\%$) of votes
  - Many rounds of counting

Example: Alternative Vote Elections in Lilliput-Blefuscu

- 100 voters
  - 40 Lilliputians (Little-endians)
  - 60 Blefuscudians (Big-endians)
- 4 candidates
  - 1 Little-endian (L)
  - 3 Big-endians
    1. Hard eggs (BH)
    2. Medium eggs (BM)
    3. Soft eggs (BS)

Example: Counting the Votes

- Counting takes place in rounds
- Each round is “last” past the post election
  1. Calculate tallies using highest preference of each ballot
  2. Exclude last candidate from counting

Example: Counting the Votes

- Counting takes place in rounds
- Each round is "last" past the post election
  1. Calculate tallies using highest preference of each ballot
  2. Exclude last candidate from counting

Candidate	L	BH	BM	BS
Round 1	40	20	25	15

Example: Counting the Votes

- Counting takes place in rounds
- Each round is "last" past the post election
  1. Calculate tallies using highest preference of each ballot
  2. Exclude last candidate from counting

Candidate	L	BH	BM	BS
Round 1	40	20	25	15
Round 2	40	25	35	-

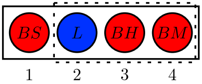
Example: Counting the Votes

- Counting takes place in rounds
- Each round is "last" past the post election
  1. Calculate tallies using highest preference of each ballot
  2. Exclude last candidate from counting

Candidate	L	BH	BM	BS
Round 1	40	20	25	15
Round 2	40	25	35	-
Round 3	40	-	60	-

Signature Attacks

- Secret ballot provides privacy and anonymity
- Signature attacks link voters to the votes they cast
  - $\Rightarrow$ Breaks receipt-freeness during the counting
  - Exploited by Italian Mafia
- Eg signed ballot with specified permutation of preferences

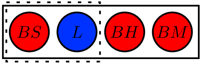


- Highly likely that randomly chosen covert signature is unique
  - Number of possible signatures is factorial in number of candidates
  - 20 candidates $\Rightarrow 19! \approx 10^{17}$ signatures

Signature Attacks on Partial Counting Information

- May still detect **absence** of some signatures
  - ⇒ Voters who disobey risk getting caught out
  - ⇒ Sufficient for bribery and coercion
- Eg round tallies reveal that some signatures never occur

Candidate	L	BH	BM	BS
Round 1	40	20	25	15
Round 2	40	25	35	-



- Increase chance of detecting absent signatures
  - Eg by embedding contrived sequences of preferences in signatures

How To Prevent Signature Attacks

- Currently no definition for what counting information enables effective signature attacks
- All information is potentially dangerous
  - ⇒ Safest approach is that counting reveals nothing apart from the result

Security Requirements for Cryptographic Counting

1. **Minimum disclosure**
   - Reveal only the identity of the winning candidate
2. Universal verifiability
   - Operations are public and accompanied by proofs
3. Robustness

Minimum Disclosure Counting Scheme

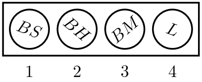
Main Idea of the Counting Scheme

1. Hide the ordering of ciphertexts
   - Mix-nets randomly permute and re-encrypt list of ciphertexts
   - Rotators randomly cyclically shift and re-encrypt list of ciphertexts
2. Seek ciphertexts with certain properties
   - Plaintext **equality**/**inequality** tests compare $[\![m_1]\!]$, $[\![m_2]\!]$
   - Tests reveal only boolean result $m_1 = m_2$ or $m_1 \geq m_2$
3. Perform open operations on identified ciphertexts
   - Eg homomorphic addition $[\![m_1]\!] \boxplus [\![m_2]\!] = [\![m_1 + m_2]\!]$

Inputs to the Counting Scheme

- Counting starts after voting finished
- Inputs:
  1. List of all candidates (encrypted and anonymous)

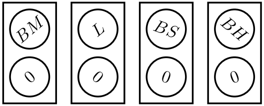
  2. List of ballots
     - Each ballot is list of encrypted preferences in decreasing order of preference



- Values encrypted with additively homomorphic cryptosystem (eg Paillier)

Tallying the Votes

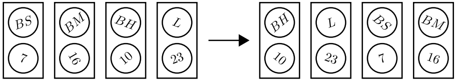
- Construct counters (encrypted candidate-tally pairs)



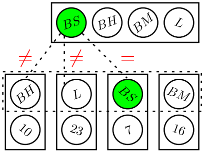
- For highest preference of each ballot, increment appropriate counter

Incrementing a Counter

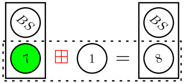
1. Mix all counters



2. Use plaintext equality tests to locate counter for *BS*

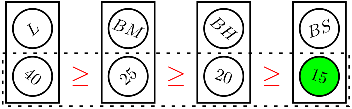


3. Openly increment tally for *BS* using homomorphic addition

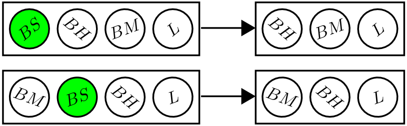


Excluding the Last Candidate

- Mix the counters
- Use plaintext inequality tests to compare encrypted tallies
  - ⇒ Minimum counter (for *BS*)

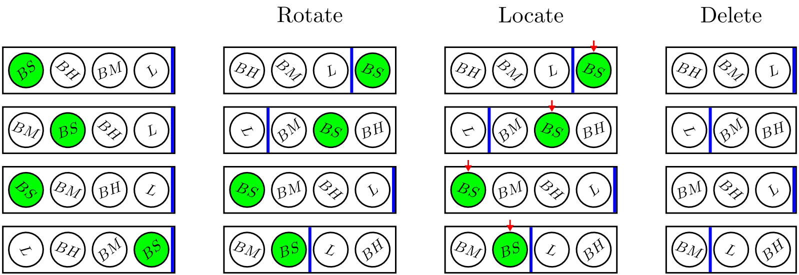


- Remove encrypted preference for *BS* from each ballot



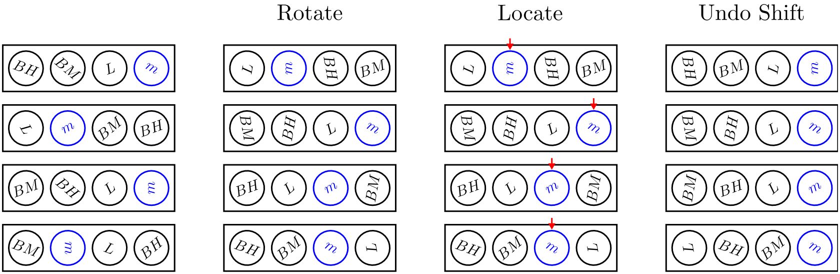
Removing the Excluded Candidate

1. Rotate all ballots to conceal positions of preferences
2. Use plaintext equality tests to locate preference for *BS*
3. Openly delete encrypted preference for *BS*



Restoring the Ballots

1. Rotate all ballots to conceal positions of deleted preferences
2. Use plaintext equality tests to locate marker
3. Openly undo cyclic shifts to return ballots to original ordering



Revealing the Winner

- Repeat rounds until only one remaining candidate
  - Constant number of rounds
- Decrypt and reveal winner

Discussion

Summary

- Signature attacks problematic for preferential counting
- Minimum disclosure property
  - Prevents signature attacks
- Minimum Disclosure Counting Scheme
  - Hide and seek paradigm preserves secrecy
- Plaintext equality and inequality tests, mix-nets, rotators
  - Provide privacy, universal verifiability and robustness
- Total complexity is $O(AC^2Vk)$

Open Problems

1. What is the optimal complexity?
    - At least $O(CV)$ distributed ballot operations
    - Limiting factor appears to be the removal of excluded candidate
    - Seems to require $O(C)$ work per ballot
2. What are the implications of weakening minimum disclosure?
    - How can we assess if specific partial counting information is sensitive?