VoteHere VHTi: A Verifiable E-Voting Protocol

Richard Barnes

Cryptography Applications Bistro,
February 3, 2004

E-Voting (in this case)

- Conducted in centralized polling places
- Ballots are electronically cast and counted
- Also known as Directly Recording Electronic (DRE) voting systems.

Verifiable E-Voting

1. Each individual voter can verify that his or her ballot as “cast-as-intended”.

3. Anyone can verify that the ballots that were cast are “counted-as-cast”.

Cast-as-Intended

- Verifiability: Assign a unique verification code to every (voter,vote) pair.
- E.g. (rand12345, Wesley Clark)

- Anonymity: Store each ballot as an encrypted form of the voting choice alone.
- E.g. E_K(Wesley Clark)

Counted-as-Cast

- Verify the Box: Publish the raw encrypted ballots with computed verification codes.
- Verify the Count: Given that the set of ballots is valid, anonymize them and count them with a secure, verifiable randomization algorithm.

Protocol: Assumptions

- Three sets of people: voters, trustees, observers.
- Trustees are trusted not to collude with each other in large numbers.
- Observers should be indistinguishable from voters (in the protocol).
- Once information has been published, it is irrevocable.

Timeline of a VHTI Election

0. Election Parameters

- Trustees set up a t of n threshold secret sharing scheme. They agree on a prime p, a secret element g of Z_p, and a public key h.
- So t of the n trustees together can read a message of the form $(g^r, h^r\ m)$.
- Assign each item on the ballot a unique identifying number α_l.

1. Generate Verification Codes

- Generate a large number of blank ballots of the form $D_i = (BSN_i, \{\alpha_l, C_i(\alpha_l)\})$ and Verification Codebook commitments.
- Verification Codes:

$$C_i(\alpha_\ell) = H\left(\alpha_\ell^{\sum_{j=1}^{n} \sigma_{ij}}\right)$$

- Commitments: Secret σ_{ij}, publish $(\gamma, \gamma^{\sigma_{ij}})$

2a. Voting and Verifying

- Get Voting Token = BSN
- Screen shows options, verification codes
- Voter makes choices
- Screen shows selected options, verification codes
- Voter gets BSN_i, α_l, $C_i(\alpha_l)$ and can verify them in published codebooks.

2b. Storing votes

- Each ballot gets stored as an encrypted pair $(g^r, h^r \alpha_l)$ with some (pseudo-)random number r_i, different for each ballot.

- These can be decrypted only by a sufficient number of trustees.

3. Counting and Verifying

- Publish encrypted ballots
- Trustees compute Verification Codes from ballots: This can be done without revealing the secrets σ_{ij}, and each trustee provides proof that his part of the computation is valid.
- Secure, verified shuffling algorithm provides anonymous, accurate count. (Developed by the author.)

Results of using VHTi

Verify cast-as-intended

- Each voter can verify that his ballot was cast as intended….
- … because the verification codes on his Ballot Receipt match those on the screen, and are verified as corresponding to his choices.

Verify Count-as-Cast

- Anyone can verify that the votes were counted as they were cast…
- …because they are provided with
  a) The encrypted ballots themselves
  b) Proof that these ballots have valid verification codes
  c) Proof that these ballots were properly anonymized and counted.

However…

- Ballot receipts open up vote-buying.
- Recall that a voter is given a BSN, vote, verification code.
- Not proof, but evidence.

However…

- Much of security rests on independent verification of proofs.
- Complexity is discouraging.
- Software to verify these proofs is vulnerable to usual problems.

Questions?

References

- C.A. Neff, J. Adler, *Verifiable e-Voting.*
- T. Pedersen. A threshold cryptosystem without a trusted party. Lecture notes in Computer Science, Springer-Verlag, 1991.
- C.A. Neff. A Verifiable Secret Shuffle and its Application to E-Voting. *Proceedings of the 8th ACM Conference on Computers and Communications Security*, 2001