Analysis of an Internet Voting Protocol

Dale Neal

Garrett Smith

Electronic Voting

- Electronic voting at a precinct
  - Focus is on preventing fraud on the part of people building and running system.
- Electronic voting over the internet
  - Must prevent fraud for all parties
  - Must provide anonymity for voters

Our chosen protocol

- *An Anonymous Electronic Voting Protocol for Voting Over The Internet*
  - Indrajit Ray, Indrakshi Ray, Natarajan Narasimhamurthi
  - University of Michigan
- Most research on internet voting focuses on new cryptographic primitives.
  - Not interesting to model at a protocol layer.

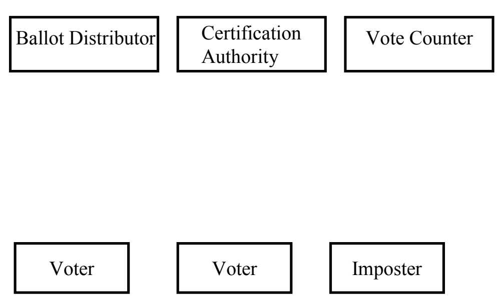
Building Blocks

- Public Key Cryptography
- Hard-to-invert permutations
- Blind Signatures on mesages

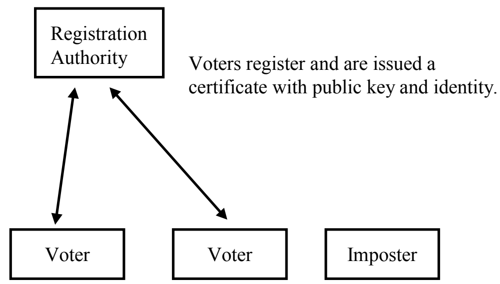
Notation

- V_e – V's encryption key
- V_d – V's decryption key (signing key)
- $[x, V_d]$ – x encrypted with V_d
- $h(x)$ – hash of x
- $\{\}$ – grouping
- $x * [b, V_e]$ – blinded submission of x for signature by V
- $[\{x * [b, V_e]\}, V_d]$ – V's blind signature of x, can be converted to $[x, V_d]$ knowing b.

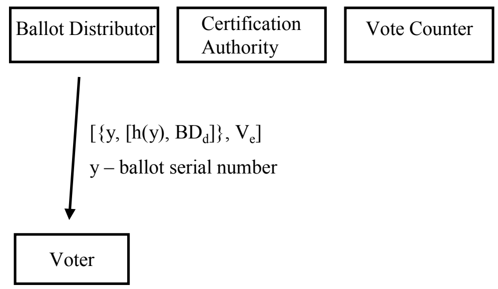
Protocol Overview



Pre-protocol setup



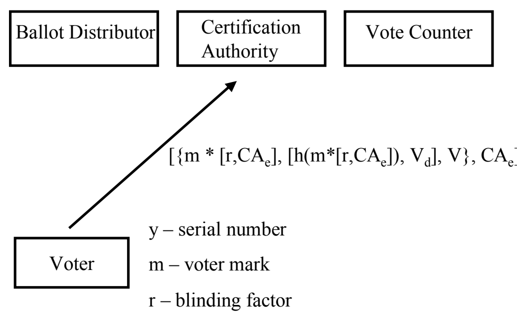
Blank Ballot Distribution



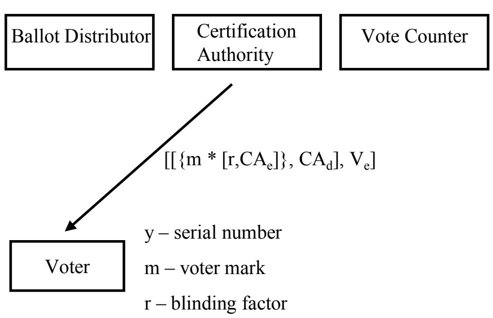
Generate a voter mark

- Voter mark allows voter to identify their ballot without letting others identify their ballot.
- Generated by a one-way permutation of the serial number.
- Poorly described in the paper
  - We assume they meant a keyed hash.

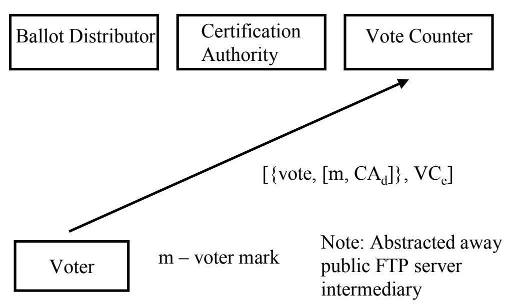
Voter Certification (part a)



Voter Certification (part b)



Vote Casting



Publishing

Ballot Distributor | Certification Authority | Vote Counter

y1
y2
y3
⋮

{m1 * [r,CA_e], [h(m1*[r,CA_e]), $V1_d$]}
{m2 * [r,CA_e], [h(m2*[r,CA_e]), $V2_d$]}
{m3 * [r,CA_e], [h(m3*[r,CA_e]), $V3_d$]}
⋮

{vote1, [m1, CA_d]}
{vote2, [m2, CA_d]}
{vote3, [m3, CA_d]}
⋮

Attack Model

- Any of CA, BD, VC could collude among themselves and with any voters.
  - Only colluding voters votes should be affected
- If fraud occurs, the fraud can be proved

Claimed Properties

- Only eligible voters are able to cast votes
- A voter is able to cast only one vote
- A voter is able to verify that his or her vote is counted in the final tally
- Nobody other than the voter can link a cast vote with a voter
- If a voter decides not to vote, nobody is able to cast a fraudulent vote in place of the voter.

Modeling in Murphi

- Encryption, signatures modeled same as in Needham-Schroeder with AgentId
- Serial number, voter mark, blind signatures modeled in the same way.
- Registered and unregistered voters
- BD, CA, VC can all act fraudulently, and accept invalid data

Invariants

- Different type of invariant than for Needham-Schroeder and other authentication protocols.
- Of the type: if there is fraud, can a party detect it?

```
invariant "voter can prove fraud if their vote is uncounted"
 forall i: GoodVoterId do
  forall j: VCId do
   voter[i].state = V_VOTED &
   multisetcount (l:vc[j].votes,
       vc[j].votes[l].signedMark = voter[i].signedMark) = 0
   ->
   ismember(voter[i].ballotSigner, BDId) &
   ismember(voter[i].markSigner, CAId)
  end
 end;
```

```
invariant "voter cannot claim fraud when they don't vote"
  forall i: GoodVoterId do
    forall j: VCId do
      voter[i].state != V_VOTED &
      multisetcount(l:vc[j].votes,
           vc[j].votes[l].signedMark = voter[i].signedMark &
           vc[j].votes[l].vote = true) = 0
       ->
       !(ismember(voter[i].ballotSigner, BDId) &
          ismember(voter[i].markSigner, CAId))
    end
  end;
```

Invariant is violated

- After Voter Certification voter has:
  - Serial number signed by BD
  - Voter mark signed by CA
- VC cannot demonstrate it never received vote as opposed to VC discarding the vote.
- Since any voter can demonstrate fraud even if none exists, demonstrations of fraud have no meaning.

Detecting know flaws

- We were able to construct an invariant to detect a flaw discussed in the paper:

  If a voter completes Voter Certification, but does not vote the three agents can collude to cast a fraudulent vote in that voters place.

```
invariant "a fraudulent vote can be detected"
  forall i: VCId do
    forall j: CAId do
      multisetcount(l:vc[i].votes,
                        vc[i].votes[l].vote = false) > 0
        ->
        multisetcount(l:vc[i].votes, true) >
              multisetcount(m:ca[j].certifications,
                              ca[j].certifications[m].response)
        -- if there is a fraudulent vote, there must
        -- be more votes than published certified voters.
      end
    end;
```

Deficiencies we couldn't model

- Ballot distribution seems unnecessary
  - Voter chooses nonce
  - CA keeps track of which voters have submitted nonces for blind signature and only signs one nonce per registered voter
- Encrypting traffic makes it harder for bystanders to eavesdrop, but doesn't provide any extra guarantees because even with CA, BD, and VC colluding they can't determine who cast what vote.

Benefits of modeling

- Ambiguities in the protocol description were cleared up by modeling the protocol and figuring out what had to be provided to ensure desired properties

Conclusions

- Being able to demonstrate fraud when there is none is a fatal flaw.
- Murphi is not well suited to modeling this flavor of protocol.
  - All of the flaws we found were discovered while trying to model the protocol
  - Proof oriented analysis seems to be a better fit
    - Prove for each type of fraud, that if it happens, then an honest party can prove that it happened