Verifiable Internet Voting for Untrusted Platforms

Mirosław Kutyłowski, Filip Zagórski

Institute of Mathematics and Computer Science
Wroclaw University of Technology

IWSEC
Nara, 29-31 X 2007

E-voting

System is based on the paper:

M. Kutylowski, F. Zagorski
Verifiable Internet Voting Solving Trusted Platform Problem

E-voting

- e-voting:
  - voting machines at polling places (expensive)
  - remote voting with electronic means

E-voting

- e-voting:
  - voting machines at polling places (expensive)
  - remote voting with electronic means
- our goal:
  - e-voting via Interent
  - easy like mail-in voting
  - ... but secure
- assumptions:
  - no PC can be trusted

Properties of mail-in voting

- excellent for vote selling
- ... or even coercing voters
- questionable anonymity (a ballot may contain a hidden code)
- ballots can disappear (Broward County, USA, 2004), or be retained until the deadline
- no verifiability

On the other hand - mail-in voting is gaining popularity!

Internet voting

A straightforward solution

- **step 0** a voter downloads a voting application from a special server,
- **step 1** using the program, the voter prepares a ballot: a resulting ciphertext that must be decrypted by many tallying authorities,
- **step 2** the voter sends a signed ballot to an appropriate authority.

- such a scenario assumes that we trust our PC

Convenient vote selling

- step 0 download and install an appropriate tracing program from a server located at *Voting Islands*,
- step 1 prepare a ballot and submit it while the tracing program is activated,
- step 2 the tracing program sends a message to Voting Islands
- step 3 the voter gets digital cash or some access codes
- step 4 the voter deinstalls the tracing program.

Countermeasures - Estonia

- the voter can cancel her electronic ballot and cast her vote in a traditional way
- so a voter may sell a vote, get money, and cancel the vote, – no reason to buy a vote

But Estonian system isn't verifiable

Breaking anonymity

- a virus can attack the voting program,
- after an attack the voting program executes “some” code,
- then somebody will be able to find out the choice of the voter without secret keys of tallying authorities,

Internet Voting - Threats Type A

- A voter can be cheated by a Voting Application (APP):
  - APP can cast a vote for a different *specified* candidate,
  - APP can cast a vote for a random candidate,
  - APP can cast an invalid vote.

Internet Voting - Threats Type B

- A voter may want to sell a vote:
    - a vote buyer/coercer is physically present when a voter is casting a vote via Internet or he can impersonate a voter,
    - a buyer/coercer provides an appropriate application to be run on a machine used for vote casting (to monitor a voter's choice)

Internet Voting - Threats Type C

- A voter can be cheated by a voting system. This problem concerns the protocols which do not provide verifiability:
  - lack of global verifiability (schemes without audit procedures)
  - lack of local verifiability

Requirements for Internet voting

A voting system must deal with all threats

1. no cheating by a PC: a PC cannot cheat the voter with high probability and manipulate the ballots
2. anonymity: the PC should not learn the choice of the voter
3. no vote selling: the voter cannot convince PC about her choice
4. verifiability: the voter can check that her vote has been counted

Decryption by multiple parties

- an El Gamal ciphertext of M for public keys $y_1, \ldots y_n$

$$(M \cdot (y_1 \cdot \ldots \cdot y_n)^k, g^k)$$

  for a random k,
- decoding requires cooperation of all holders of private keys corresponding to $y_1, \ldots y_n$, decoding must be performed as follows

$$(c_1, c_2) := (c_1 / c_2^{x_i}, c_2)$$

  for $i = 1, 2, \ldots$

Re-encryption

knowing public keys $y_1, \ldots, y_n, g$, one can re-encrypt a ciphertext

$$(a, b) = (m \cdot (y_1 \cdot \ldots \cdot y_n)^{k_i}, g^{k_i})$$

of m by choosing, at random k_{i+1} and computing:

$$(a \cdot (y_1 \cdot \ldots \cdot y_n)^{k_{i+1}}, b \cdot g^{k_{i+1}})$$

which yields:

$$\begin{aligned} & (m \cdot (y_1 \cdot \ldots \cdot y_n)^{k_i} \cdot (y_1 \cdot \ldots \cdot y_n)^{k_{i+1}}, g_i^k \cdot g^{k_{i+1}}) = \\ = \quad & (m \cdot (y_1 \cdot \ldots \cdot y_n)^{k_i + k_{i+1}}, g^{k_i + k_{i+1}}) \end{aligned}$$

Pairs & cut and choose

- a ballot for X: four randomly permuted ciphertexts prepared:

$$v_{X,r} = \quad [c(X) \quad c(r') \quad c(X') \quad c(r)]$$

  two matching pairs:
  - a pair of ciphertexts encoding vote (X, X')
  - a pair of ciphertexts encoding vote identifier (r, r')
- voter knows value of r
- APP does not know which ciphertext encodes r, which encodes X
- to count vote for X - pair (X, X') has to appear on the last bulletin board

Our solution - high level scenario

- a voter's PC is running a Voting Application (APP)
- a Ballot Generation Server (SERVER) – prepares cards
- a Registration Server (RS) – checks voter's identity and inserts ballots into the system
- Mixes - decode votes (Mix's output is published on public Bulletin Board)
- on the last Bulletin Board an election results will appear

Equipment

- a voter has her personal smart card, the same which is used by her for advanced electronic signature
- a voter (Alice) will not give/lend her card to vote buyer, because he can sign e. g. “I sell my car for 1$. Alice”
- such cards are gaining popularity, e. g. in Estonia are used as ID cards

Applied tricks

- commitments and cut & choose games – prevents cheats by the SERVER and APP
- independent channel (phone line, SMS, mail) – prevents cheating by APP
- re-encryption helps keeping voter's choice secret
- vote revoking – prevents vote selling

A voting card

- there is publicly known base ordering of list of candidates: X_0, X_1, X_2
- a voting card shifted by $s_1 = 1, s_2 = 1$, $s = s_1 + s_2 \bmod 3 = 2$:

$$[X_1]$$

$$[X_2]$$

$$[X_0]$$

A voting card

- there is publicly known base ordering of a list of candidates: X_0, X_1, X_2
- a voting card shifted by $s = s_1 + s_2 \bmod 3 = 2$:

$$[c(X_1)]$$

$$[c(X_2)]$$

$$[c(X_0)]$$

- where $c(X_i)$ stands for cryptogram of X_i

A voting card

- a voting card (shift $s = 2$, identifier r, $\overline{A}A\overline{A}$):

$$
\begin{array}{ccccc}
[c(r) & c(X_1) & c(r') & c(X_1')] & c(A) \\
[c(X_2') & c(X_2) & c(r') & c(r)] & c(\overline{A}) \\
[c(X_0) & c(r') & c(X_0') & c(r)] & c(\overline{A})
\end{array}
$$

A voting card

- a voting card (shift $s = 2$, identifier r, $\overline{A}A\overline{A}$):

$$
\begin{array}{ccccc}
[c(r) & c(X_1) & c(r') & c(X_1')] & c(A) \\
[c(X_2') & c(X_2) & c(r') & c(r)] & c(\overline{A}) \\
[c(X_0) & c(r') & c(X_0') & c(r)] & c(\overline{A})
\end{array}
$$

- it has to be generated by SERVER (RE-RSA signatures attached to a voting card later)

A voting card

- a voting card (shift $s = 2$, identifier r, $\overline{A}A\overline{A}$):

$$\begin{array}{ll}
[c(r) \quad c(X_1) \quad c(r') \quad c(X_1')] & c(A) \\
[c(X_2') \quad c(X_2) \quad c(r') \quad c(r)] & c(\overline{A}) \\
[c(X_0) \quad c(r') \quad c(X_0') \quad c(r)] & c(\overline{A})
\end{array}$$

- it has to be generated by SERVER (RE-RSA signatures attached to a voting card later)
- after re-encryption and permutation of a row performed by APP, SERVER does not know which row of the card was chosen
- voter knows r and is convinced about a values of s_1, s_2 (and thus knows s)

Attempts of manipulation

- successful vote replacement:
  - one have to find a pair which encodes a vote (and not an identifier)
  - removing only a half of a vote or identifier can be detected!
  - it is unlikely to find the matching halves if all ciphertexts are mixed

Voting process

- SERVER sends k voting cards with commitments
- verification is made by APP & Voter

Voting process

- SERVER sends *k* voting cards with commitments
- verification is made by APP & Voter
- voter chooses her candidate
- APP casts voter's ballot to the first Mix
- voter can verify if her vote is correct

Voting process

- SERVER sends k voting cards with commitments
- verification is made by APP & Voter
- voter chooses her candidate
- APP casts voter's ballot to the first Mix
- voter can verify if her vote is correct
- tallying begins (mixing & decrypting)
- Randomized Partial Checking (RPC)

Voting cards generation (simple)

- SERVER generates k voting cards and sends them to APP together with commitments
- APP asks SERVER to reveal all but one card
  - choice is made deterministically – based on the signature made by smart card under publicly known value
  - voter can know it in advance
- APP verifies unused cards
- SERVER has $\frac{1}{k}$ chance of undetected cheating
- SERVER sends to APP appropriate signatures of a chosen card

Voting cards generation

- SERVER generates k voting cards and sends them to APP
- together with voting cards
  - “some” commitments are sent to APP
  - voter obtains (phone line, mail, SMS):
    - short voting cards identifiers
    - shifts s_1, s_2 used in voting cards (and commiments to s_1, s_2)

Voting cards generation

- SERVER generates k voting cards and sends them to APP
- together with voting cards
  - "some" commitments are sent to APP
  - voter obtains (phone line, mail, SMS):
    - short voting cards identifiers
    - shifts s_1, s_2 used in voting cards (and commiments to s_1, s_2)
  - voter can check on the Webpage (short voting card id)
    - voting card identifiers (r)
    - marks $A, \overline{A}$, sequence T_A
    - signature of the SERVER
  - APP presents to a votes values revealed during verification – voter is convinced that everything is OK

Casting a vote

- voter knows shift s used in a voting card
- voter likes candidate i
- voter chooses candidate $s+i \bmod K$ (K - number of candidates)
- APP:
  - chooses $(s+i \bmod K)^{th}$ row of a voting card - ballot
  - re-encrypts ballot
  - signs it together with smart card
  - sends it to Registration Server (RS)

Verification of vote casting

- RS publishes decrypted mark $A/\overline{A}$
- voter checks if a mark is correct

Tallying

- Ballots are mixed and decrypted by Mix servers
- every part of a ballot goes separately through Mix-cascade
- results appear on final Bulletin Board
- RPC

Cancelling procedure

- a voter casts a vote together with an *anti-vote*
  - a vote is re-encrypted and appears on the first Bulletin Board,
  - an encrypted anti-vote is kept by the Registration Server

Cancelling procedure

- a voter casts a vote together with an *anti-vote*
  - a vote is re-encrypted and appears on the first Bulletin Board,
  - an encrypted anti-vote is kept by the Registration Server
- the previous vote can be cancelled by Registration Server only with cooperation with a voter (a voter must sign an appropriate message)
- vote and anti-vote are unlinkable

Voting Application APP

- everyone can write his own APP or use one from independent, trusted third party
- APP gains no information about card properties i. e. shift used in a voting card
- even if Voter tells APP about shift, she can revoke her vote and vote again
- APP cannot vote for randomly chosen candidate (will be caught with probability $\geq \frac{1}{2}$
- APP cannot vote for specified candidate because does not know shift used

Ballot Generating Server (SERVER)

- for every voter, SERVER sends $k \geq 2$ voting cards and commitments to the values used for generating cards
- a voter chooses one of the cards and verifies the rest - SERVER reveals values and a Voter together with APP checks if a cards are properly encoded

$\Rightarrow$ a voter knows (with probability $1 - \frac{1}{k}$) that a card which she will use is proper

Voting process

- SERVER sends k voting cards with commitments
- verification is made by APP & Voter

$$1 - \frac{1}{k}$$

Voting process

- SERVER sends k voting cards with commitments
- verification is made by APP & Voter
$$1 - \frac{1}{k}$$
- voter chooses her candidate
- APP casts voter's ballot to the first Mix
- voter can verify if her vote is correct
$$1 - \frac{1}{2^l}$$

Voting process

- SERVER sends k voting cards with commitments
- verification is made by APP & Voter
$$1 - \frac{1}{k}$$
- voter chooses her candidate
- APP casts voter's ballot to the first Mix
- voter can verify if her vote is correct
$$1 - \frac{1}{2^l}$$
- tallying begins (mixing & decrypting)
- RPC $\quad 1 - \frac{1}{2^m}$

Main features - summary

- coercion-freeness
- global verifiability (randomized partial checking on mixing process, ballots correctly decoded)
- local verifiability - every voter has “end-to-end” verifiability (in other schemes voter’s verification ends on the first Bulletin Board)
- we do not demant a voter’s PC to be trusted

Thank you for your attention!