University of Fribourg

Bern University of Applied Sciences

Choosing a Code Verification Protocol

Oliver Spycher

Biel, April 23rd 2012

Outline

Motivation, History and Overview

Propositions
- Gjosteen 2010
- Lipmaa 2011 first
- Lipmaa 2011 second

Comparison and Conclusions

Outline

Secure Platform Problem

What if the adversary is in control of voters' platforms.

David Chaum's Code Voting 2001

- Voters obtain a code sheet
- Cast a code per candidate
- Verify another code per candidate

Jörn Helbach's Optimization 2007

- Verify another code per candidate (Confirmation TAN)
- Send a message of acceptance (Finalization TAN)

What problems are solved / left open?

Practicability

Isn't i-voting supposed to be easy?

Use just confirmation TAN

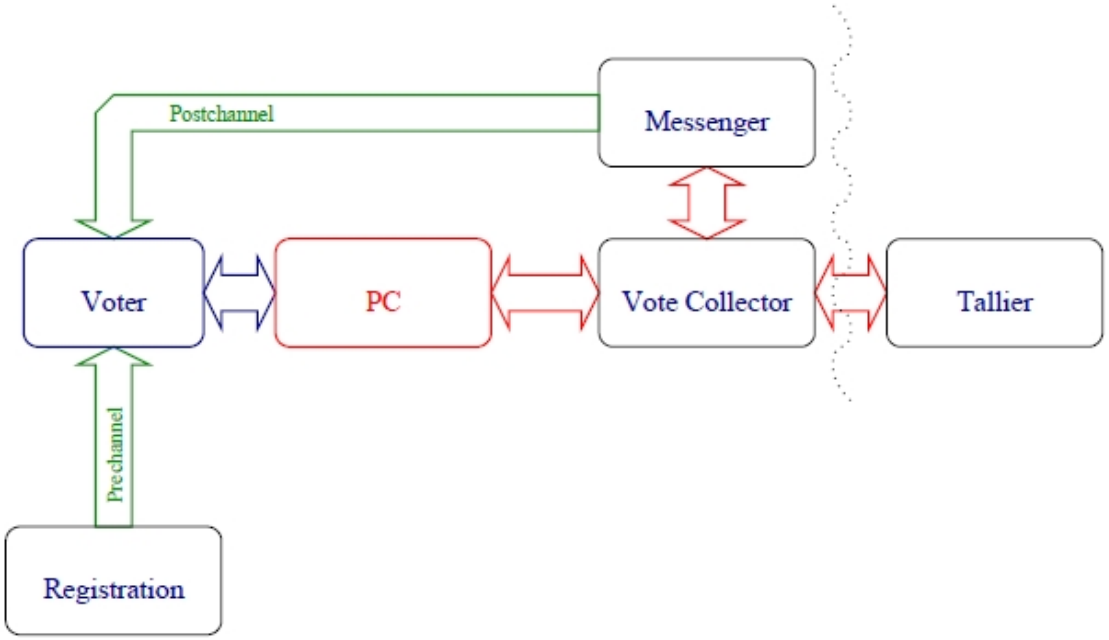
- Click your candidates
- Verify one code per candidate

Propositions for the Real World

- Gjosteen 2010 (Norway's solution)
- Heiberg, Lipmaa, Van Laenen 2010
- Puiggali, Guasch 2011 (Scytl's orig. proposal for Norway)
- Lipmaa (2 propositions) 2011

Re-voting required by tradition in Norway. $\longrightarrow$ Implications?

Common Setting



Outline

Gjosteen: Corruption Model and Security Goal

the attacker may corrupt either..

1. the vote collector and any subset of voters and computers
2. or any infrastructure player

→ auditor required to verify computations of corrupted players

security goal

1. the usual integrity goals or failure
2. privacy when using honest computers

Prerequisites

This is simplified!

Safe ElGamal p, q, g and signing mechanism
Value $c \in \mathcal{G}_q$ per candidate (let c identify the candidate)

TTP chooses $d_1 + d_2 \equiv d_3 \mod q$; $d_i \in \mathbb{Z}_q$

- d_1 private key of tallier ($e_1 = g^{d_1}$)
- d_2 private key of vote collector ($e_2 = g^{d_2}$)
- d_3 private key of messenger ($e_3 = g^{d_3}$)

TTP chooses random $s_v \in \mathbb{Z}_q$ per voter

- sends (c, c^{s_v}) to voter for each c (code sheet)

Voting and Verification (1)

Voter sends choice $(c_1, .., c_k)$ to his computer

the computer...

- chooses randomness $r_i \in \mathbb{Z}_q$
- computes all $(x_i, y_i) = (g^{r_i}, e_1^{r_i} \cdot c_i)$
- $\rightarrow$ computes ZKP of knowledge of plaintexts c_i
- $\rightarrow$ computes signature
- sends (x_i, y_i), signature and ZKP to vote collector

$\rightarrow$ why ZKP, why signature?

Voting and Verification (2)

the vote collector..

- verifies computer proof and signature
- computes all $\bar{x}_i = x_i^{s_v}$
- computes all $\bar{y}_i = y_i^{s_v} \cdot \bar{x}_i^{d_2}$
- $\longrightarrow$ computes ZKP to prove correct computation
- signs and sends all $(\bar{x}_i, \bar{y}_i)$, all proofs and the signature to messenger

$\longrightarrow$ why ZKP?

Voting and Verification (3)

the messenger..

- verifies signature and both proofs
- computes all $c_i^{s_v}$ as $Dec_{d_3}((\bar{x}_i, \bar{y}_i))$
- $\longrightarrow$ computes and signs hash of all (x_i, y_i)
- sends signature and hash to voter's computer through vote collector (who then permanently stores vote)
- sends SMS to voter containing all $c_i^{s_v}$ (receipt)

$\longrightarrow$ Why hash and signature?

Voting and Verification (4)

the computer..

- verifies messenger's signature using all (x_i, y_i)
- suggests success to voter

the voter..

- acknowledges computer's success message
- verifies occurrence of $(c_i, c_i^{s_v})$ in his code sheet
- can re-vote in case of doubts

Counting

the tallier..

- receives all votes to be counted from vote collector
- mixes and decrypts the votes
- generates ZKP of correct computation
- sends decrypted vote and ZKP to auditor

the auditor..

- uses full vote collector contents and hashes from messenger to audit input to tallier
- verifies ZKP from tallier

How can vote collector and messenger collude to break privacy?

How can vote collector and messenger collude to break privacy?

- Compute the tallier's private key $d_1 \leftarrow d_3 - d_2$ / What else?

How can vote collector and messenger collude to break privacy?

- Compute the tallier's private key $d_1 \leftarrow d_3 - d_2$ / What else?
- Establish map (c, c^{s_v})

How can vote collector and messenger collude to break privacy?

- Compute the tallier's private key $d_1 \leftarrow d_3 - d_2$ / What else?
- Establish map (c, c^{s_v})

Lipmaa doesn't want online components to break privacy

- His first proposal solves the first problem.
- His second additionally solves the second problem

Changes to Setup - before

TTP chooses $d_1 + d_2 \equiv d_3 \mod q$; $d_i \in \mathbb{Z}_q$

- d_1 private key of tallier ($e_1 = g^{d_1}$)
- d_2 private key of vote collector ($e_2 = g^{d_2}$)
- d_3 private key of messenger ($e_3 = g^{d_3}$)

TTP chooses random $s_v \in \mathbb{Z}_q$ per voter

- sends (c, c^{s_v}) to voter for each c (code sheet)

Lipmaa's first: Changes to Setup

TTP chooses $d_1, d_3 \mod q$; $d_i \in \mathbb{Z}_q$

- d_1 private key of tallier ($e_1 = g^{d_1}$)
-
- d_3 private key of messenger ($e_3 = g^{d_3}$)

TTP chooses random $s_v \in \mathbb{Z}_q$ per voter and symmetric k

- sends secret s_v, k and $h_v = g^{s_v}$ to voter
- sends $(c, h_v^{AES_k(c)})$ to voter for each c (code sheet)
- sends $(c, g^{AES_k(c)})$ to tallier for each c

Voting and Verification (1) - before

Voter sends choice $(c_1, .., c_k)$ to his computer

the computer...

- chooses randomness $r_i \in \mathbb{Z}_q$
- computes all $(x_i, y_i) = (g^{r_i}, e_1^{r_i} \cdot c_i)$
- $\longrightarrow$ computes ZKP of knowledge of plaintexts c_i
- $\longrightarrow$ computes signature
- sends (x_i, y_i), signature and ZKP to vote collector

Voting and Verification (1)

Voter sends choice $(c_1, .., c_k)$ to his computer

the computer...

- chooses randomnesses $r_i, R_i \in \mathbb{Z}_q$
- computes all $(x_i, y_i) = (g^{r_i}, e_1^{r_i} \cdot g^{AES_k(c_i)})$
- computes all $(X_i, Y_i) = (g^{R_i}, e_3^{R_i} \cdot h_v^{AES_k(c_i)})$
- $\rightarrow$ computes ZKP of knowledge of r_i, R_i, s_v and $AES_k(c_i)$
- $\rightarrow$ computes signature
- sends (x_i, y_i), (X_i, Y_i), h_v, signature and ZKP to vote collector

Voting and Verification (2) - before

the vote collector..

- verifies computer proof and signature
- computes all $\bar{x}_i = x_i^{s_V}$
- computes all $\bar{y}_i = y_i^{s_v} \cdot \bar{x}_i^{d_2}$
- $\longrightarrow$ computes ZKP to prove correct computation
- signs and sends all $(\bar{x}_i, \bar{y}_i)$, all proofs and the signature to messenger

Voting and Verification (2)

the vote collector..

- verifies computer proof and signature
- signs and sends all (x_i, y_i), (X_i, Y_i), h_v, signature and ZKP to messenger

Voting and Verification (3) - before

the messenger..

- verifies signature and both proofs
- computes all $c_i^{s_v}$ as $Dec_{d_3}((\bar{x}_i, \bar{y}_i))$
- $\longrightarrow$ computes and signs hash of all (x_i, y_i)
- sends signature and hash to voter's computer through vote collector (who then permanently stores vote)
- sends SMS to voter containing all $c_i^{s_v}$ (receipt)

Voting and Verification (3)

the messenger..

- verifies signatures and proof
- computes all $h_v^{AES_k(c_i)}$ as $Dec_{d_3}((X_i, Y_i))$
- (further steps omitted)

How can vote collector and messenger collude to break privacy?

How can vote collector and messenger collude to break privacy?

- ~~Compute the tallier's private key $d_1 \leftarrow d_3 - d_2$ / What else?~~
- Establish map (c, c^{s_v}) - This still works in analogy, since h_v needs to be known. (However for the attack to work, it is assumed that k is known.)

Lipmaa doesn't want online components to break privacy

- Now we make modifications to obtain his second proposal

Vote Casting - before

Voter sends choice $(c_1, .., c_k)$ to his computer

the computer...

- chooses randomnesses $r_i, R_i \in \mathbb{Z}_q$
- computes all $(x_i, y_i) = (g^{r_i}, e_1^{r_i} \cdot g^{AES_k(c_i)})$
- computes all $(X_i, Y_i) = (g^{R_i}, e_3^{R_i} \cdot {h_v}^{AES_k(c_i)})$
- ....
- sends (x_i, y_i), (X_i, Y_i), h_v, signature and ZKP to vote collector

Lipmaa's second: Vote Casting

Additional values

- Additional public value $h \in \mathrm{G}_q$ generated at setup
- Voter sends pedersen commitment C_v to s_v as $g^{s_v} \cdot h^{z_v}$ instead of h_v

Voter sends choice $(c_1, .., c_k)$ to his computer

the computer...

- $\longrightarrow$ computes ZKP of knowledge of r_i, R_i s_v, z_v and $AES_k(c_i)$
- $\longrightarrow$ computes signature
- sends (x_i, y_i), (X_i, Y_i), C_v, signature and ZKP to vote collector

Outline

Performance

Protocol	Voter PC	Vote Collector	Messenger	Setup phase
[HLV10]	$(7\gamma + 10) \cdot e + 1 \cdot s$	$(2\Gamma + 6\gamma + 8) \cdot e + 1 \cdot v + 1 \cdot s$	$\Gamma \cdot e + 1 \cdot v$	No
[Gjø10]	$3 \cdot e + 1 \cdot s$	$8 \cdot e + 1 \cdot v + 1 \cdot s$	$10 \cdot e + 1 \cdot v$	Yes
Sect. 4	$12 \cdot e + 1 \cdot s$	$9 \cdot e + 1 \cdot v + 1 \cdot s$	$10 \cdot e + 2 \cdot v$	Yes
Sect. 5	$16 \cdot e + 1 \cdot s$	$17 \cdot e + 1 \cdot v + 1 \cdot s$	$18 \cdot e + 2 \cdot v$	Yes

Conclusions

Science is aiming for practicable solutions

- to solve SPP
- to keep i-voting user-friendly

At the cost of efficiency Lipmaa's second proposal

- keeps online backend components from breaking privacy (mind the doubt)
- shifts trust assumptions to implicit components

What about CH?