technology
from seed

An Efficient and Highly Sound Voter Verification Technique (MarkPledge 3) and its Implementation

Rui Joaquim rjoaquim@cc.isel.ipl.pt (Inesc-ID / IPL)
Carlos Ribeiro carlos.ribeiro@ist.utl.pt (Inesc-ID / UTL)

inesc id lisboa
Distributed Systems Group

VoteID 2011, Tallinn, Estonia, 28-30 September 2011

Introduction

- Main goal of electronic voting research:
  - How to create and deploy an electronic voting system that protects the voter’s privacy and outputs verifiable results?

Introduction

- Main goal of electronic voting research:
  - How to create and deploy an electronic voting system that protects the voter's privacy and outputs verifiable results?
- Two verifiability problems to address:
  - Counted-as-cast verification
    - The tally is the correct sum of the casted votes
  - Cast-as-intended verification
    - The vote cast into the system represents the voter's choice

Contribution and Motivation

- **Our contribution, MarkPledge 3**:
  - MarkPledge 3 is a new voter verifiable encryption technique to ensure the voter that her vote intention is correctly recorded by the voting system.

    (**a new technique to provide cast-as-intended verification**)

Contribution and Motivation

- **Our contribution, MarkPledge 3**:
  - MarkPledge 3 is a new voter verifiable encryption technique to ensure the voter that her vote intention is correctly recorded by the voting system.

    (**a new technique to provide cast-as-intended verification**)

- **Motivation**
  - We wanted to develop a new verifiable Internet voting system that could run on constrained environments, e.g. smart cards and secure elements inside smart phones.
  - We have chosen the MarkPledge voter verifiable vote encryption technique for our cast-as-intended verification mechanism.
    - The MarkPledge technique actually verifies the casted vote and not a test vote or a pre-encrypted ballot.
  - But, the previous MarkPledge specifications do not perform fast enough.

Vote Encryption
Implementation results overview

- Vote encryption times for a ballot with 10 candidates

	MP1	MP2	MP3
JavaCard (|p|=1024, |q|=512)	8.5 min	15 hours	1.5 min
MULTOS card (|p|=1024, |q|=512)	5 min	30 min	43 sec
MULTOS card (|p|=1024, |q|=160)	4 min	2.8 min	28 sec

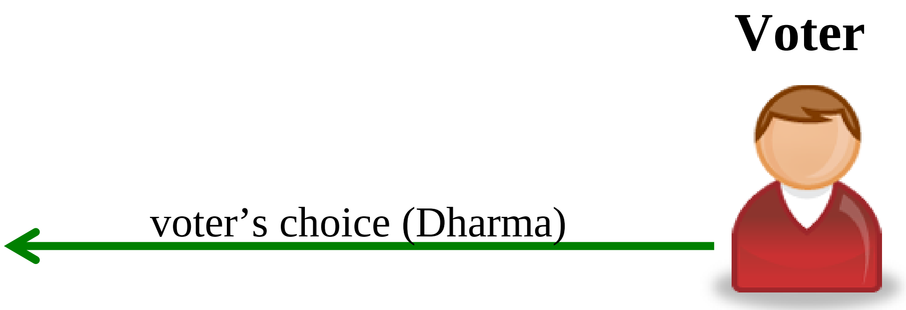
p and q are cryptosystem parameters (ElGamal parameters)

MarkPledge verification technique overview

MarkPledge Overview

Vote Encryption

Candidate	Cand. Encryption
Alice	NO 8FD3
Bob	NO IRN1
Charles	NO 72T9
Dharma	YES PZ8R



technology
from seed

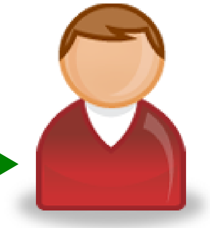
inesc id lisboa

Vote Encryption

Candidate	Cand. Encryption
Alice	NO 8FD3
Bob	NO IRN1
Charles	NO 72T9
Dharma	YES PZ8R

value disclosed (pledged) only to the voter

Voter



MarkPledge Overview

technology
from seed

inesc id lisboa

Vote Encryption

Candidate	Cand. Encryption
Alice	NO 8FD3
Bob	NO IRN1
Charles	NO 72T9
Dharma	YES PZ8R

Random Challenge

$\mathcal{R}C_{pk}$

Vote Receipt

Candidate	Verification code
Alice	46R9
Bob	QE41
Charles	KNSY
Dharma	PZ8R

MarkPledge Overview

Vote Encryption

Candidate	Cand. Encryption
Alice	NO 8FD3
Bob	NO IRN1
Charles	NO 72T9
Dharma	YES PZ8R

Random Challenge

$\mathcal{R}C_{pk}$

Vote Receipt

Candidate	Verification code
Alice	46R9
Bob	QE41
Charles	KNSY
Dharma	PZ8R

MarkPledge Overview

Vote Encryption

Candidate	Cand. Encryption
Alice	NO 8FD3
Bob	NO IRN1
Charles	NO 72T9
Dharma	YES PZ8R

Random Challenge

$\mathcal{R}C_{pk}$

Voter

Vote Receipt

Candidate	Verification code
Alice	46R9
Bob	QE41
Charles	KNSY
Dharma	PZ8R

YES vote verification **soundness = $1\text{-}2^{-\alpha}$**
(α = 24 => $1\text{-}2^{-24}$ = 0,99999994)

The voter verifies that the verification code it is the pledged value.

MarkPledge details
Preliminaries

MarkPledge specifications use the ElGamal over the subgroup G_q of $\mathbb{Z}_p^*$, where g is generator of G_q and p, q are large primes such that $q \mid p-1$.

Private key is x: $0 < x < q$

Public key is $h = g^x$

Exponential ElGamal encryption of $m = \mathcal{E}_h(m,r) = \langle g^r, h^r \cdot g^m \rangle$,
where $0 < r < q$ and $m \in \mathbb{Z}_q$

MarkPledge specifications take advantage of the following homomorphisms:

$\mathcal{E}_h(m_1,r_1) \cdot \mathcal{E}_h(m_2,r_2) = \mathcal{E}_h(m_1 + m_2, r_1 + r_2)$ (additive)

$\mathcal{E}_h(m,r)^n = \mathcal{E}_h(m \cdot n, r \cdot n)$ (multiplicative)

Previous MarkPledge solutions

- MarkPledge 1 (Neff, 2004)
  - Two ciphertexts per receipt verification code bit:
    - 24 bits -> 48 ciphertexts per candidate
  - Working principle:
    - 2-out-of-1 cut-and-choose in each verification code bit.
- MarkPledge 2 (Adida and Neff, 2009)
  - Two encryptions:
    - Encrypts the coordinates of a 2D vector (2 ciphertexts per candidate)
  - Working principle:
    - Vector algebra between special classes of 2D vectors (vector dot product).
    - Vector classes defined over the SO(2,q) of matrices defined based on the ElGamal parameters.
    - Requires modular matrix exponentiations.

      (no direct hardware acceleration for matrix exponentiations)

MarkPledge 3 Details

MarkPledge 3
Vote encryption and verification details

Vote Machine | **Voter** | **Third Parties**

$b' \in \{-1_{(NO)}, 1_{(YES)}\}$

$\tau, \delta, \theta \in_R \mathbb{Z}_q$

$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$

$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$

$\xrightarrow{u, v}$

$\xrightarrow{pledge = \theta}$

MarkPledge 3
Vote encryption and verification details

Vote Machine		Voter	Third Parties
$b' \in \{-1_{(NO)}, 1_{(YES)}\}$			
$\tau, \delta, \theta \in_R \mathbb{Z}_q$			
$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$			
$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$	u, v →		
	$pledge = \theta$ →		
	← c	$c \in_R \mathbb{Z}_q$	

MarkPledge 3
Vote encryption and verification details

Vote Machine | **Voter** | **Third Parties**

$b' \in \{-1_{(NO)}, 1_{(YES)}\}$

$\tau, \delta, \theta \in_R \mathbb{Z}_q$

$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$

$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$ $\xrightarrow{u, v}$

$\xrightarrow{pledge = \theta}$

$\xleftarrow{c}$ $c \in_R \mathbb{Z}_q$

$\vartheta \leftarrow \dfrac{b' \cdot c - c + \theta}{b'}$

$\omega \leftarrow \tau \cdot (c - \vartheta) + \delta$ $\xrightarrow{\vartheta, \omega}$

MarkPledge 3
Vote encryption and verification details

technology
from seed

inesc id lisboa

Vote Machine | **Voter** | **Third Parties**

$b' \in \{-1_{(NO)}, 1_{(YES)}\}$

$\tau, \delta, \theta \in_R \mathbb{Z}_q$

$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$

$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$ $\xrightarrow{u, v}$

$\xrightarrow{pledge = \theta}$

$\xleftarrow{c}$ $c \in_R \mathbb{Z}_q$

$\vartheta \leftarrow \frac{b' \cdot c - c + \theta}{b'}$

$\omega \leftarrow \tau \cdot (c - \vartheta) + \delta$ $\xrightarrow{\vartheta, \omega}$

$pledge_{(YES)} \overset{?}{=} \vartheta$

$pledge_{(NO)} \overset{?}{=} 2 \cdot c - \vartheta$

MarkPledge 3
Vote encryption and verification details

Vote Machine		Voter		Third Parties
$b' \in \{-1_{(NO)}, 1_{(YES)}\}$				
$\tau, \delta, \theta \in_R \mathbb{Z}_q$				
$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$				
$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$	$\xrightarrow{u, v}$			
	$\xrightarrow{pledge = \theta}$			
	$\xleftarrow{c}$	$c \in_R \mathbb{Z}_q$		
$\vartheta \leftarrow \dfrac{b' \cdot c - c + \theta}{b'}$				
$\omega \leftarrow \tau \cdot (c - \vartheta) + \delta$	$\xrightarrow{\vartheta, \omega}$		$\xrightarrow{u, v, c, \vartheta, \omega}$	
		$pledge_{(YES)} \stackrel{?}{=} \vartheta$		$u^{c-\vartheta} \cdot v \stackrel{?}{=} \langle g^{\omega}, h^{\omega} \cdot g^{c} \rangle$
		$pledge_{(NO)} \stackrel{?}{=} 2 \cdot c - \vartheta$		

MarkPledge 3
Vote encryption and verification details

technology
from seed

inesc id lisboa

Vote Machine | **Voter** | **Third Parties**

$b' \in \{-1_{(NO)}, 1_{(YES)}\}$

$\tau, \delta, \theta \in_R \mathbb{Z}_q$

$u \leftarrow \langle g^{\tau}, h^{\tau} \cdot g^{b'} \rangle$

$v \leftarrow \langle g^{\delta}, h^{\delta} \cdot g^{\theta} \rangle$ $\xrightarrow{u,v}$

$\xrightarrow{pledge = \theta}$

$\xleftarrow{c}$ $c \in_R \mathbb{Z}_q$

$$\vartheta \leftarrow \frac{b' \cdot c - c + \theta}{b'}$$

$\omega \leftarrow \tau \cdot (c - \vartheta) + \delta$ $\xrightarrow{\vartheta, \omega}$

$\xrightarrow{u, v, c, \vartheta, \omega}$

$pledge_{(YES)} \stackrel{?}{=} \vartheta$

$pledge_{(NO)} \stackrel{?}{=} 2 \cdot c - \vartheta$

$u^{c-\vartheta} \cdot v \stackrel{?}{=} \langle g^{\omega}, h^{\omega} \cdot g^{c} \rangle$

MarkPledge 3
Third party verification

technology
from seed

$$u^{c-\vartheta} \cdot v = \langle g^{\omega}, h^{\omega} \cdot g^{c} \rangle$$

MarkPledge 3
Third party verification

technology
from seed

$$u^{c-\vartheta}\cdot v = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Leftrightarrow \langle g^{\tau}, h^{\tau}\cdot g^{b'}\rangle^{c-\vartheta}\cdot\langle g^{\delta}, h^{\delta}\cdot g^{\theta}\rangle = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Rightarrow$$

$$b'\cdot(c-\vartheta)+\theta = c \pmod q$$

θ – random commit code

c – challenge

ϑ – verification code

ω – validation factor

MarkPledge 3
Third party verification

technology from seed

inesc id lisboa

$$u^{c-\vartheta}\cdot v=\langle g^{\omega},h^{\omega}\cdot g^{c}\rangle \Leftrightarrow \langle g^{\tau},h^{\tau}\cdot g^{b'}\rangle^{c-\vartheta}\cdot\langle g^{\delta},h^{\delta}\cdot g^{\theta}\rangle = \langle g^{\omega},h^{\omega}\cdot g^{c}\rangle \Rightarrow$$

$$b'\cdot(c-\vartheta)+\theta=c \pmod q$$

θ – random commit code

c – challenge

ϑ – verification code

ω – validation factor

- **YES (b' = 1)**

$$1\cdot(c-\vartheta)+\theta=c \Leftrightarrow \vartheta=\theta$$

The verification code is a "pledgeable" value ($\vartheta = \theta$)

MarkPledge 3
Third party verification

technology
from seed

$$u^{c-\vartheta}\cdot v = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Leftrightarrow \langle g^{\tau}, h^{\tau}\cdot g^{b'}\rangle^{c-\vartheta}\cdot\langle g^{\delta}, h^{\delta}\cdot g^{\theta}\rangle = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Rightarrow$$

$$b'\cdot(c-\vartheta)+\theta = c \pmod q$$

$\theta -$ random commit code

$c -$ challenge

$\vartheta -$ verification code

$\omega -$ validation factor

- YES (b' = 1)

  $$1\cdot(c-\vartheta)+\theta = c \Leftrightarrow \vartheta = \theta$$

  The verification code is a "pledgeable" value (ϑ = θ)

- **NO (b' = -1)**

  $$-1\cdot(c-\vartheta)+\theta = c \Leftrightarrow \vartheta = 2\cdot c-\theta$$

  The verification code is NOT a "pledgeable" value (ϑ = 2.*c* – θ)

MarkPledge 3
Third party verification

technology from seed

inesc id lisboa

$$u^{c-\vartheta}\cdot v = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Leftrightarrow \langle g^{\tau}, h^{\tau}\cdot g^{b'}\rangle^{c-\vartheta}\cdot\langle g^{\delta}, h^{\delta}\cdot g^{\theta}\rangle = \langle g^{\omega}, h^{\omega}\cdot g^{c}\rangle \Rightarrow$$

$$b'\cdot(c-\vartheta)+\theta = c \pmod q$$

θ – random commit code

c – challenge

ϑ – verification code

ω – validation factor

- YES (b' = 1)

$$1\cdot(c-\vartheta)+\theta = c \Leftrightarrow \vartheta = \theta$$

The verification code is a "pledgeable" value ($\vartheta = \theta$)

- NO (b' = -1)

$$-1\cdot(c-\vartheta)+\theta = c \Leftrightarrow \vartheta = 2\cdot c-\theta$$

The verification code is NOT a "pledgeable" value ($\vartheta = 2.c - \theta$)

Conclusion:

If the voter verifies that pledge = ϑ it is guaranteed, with a very high soundness, that the voter has found a YES vote encryption.

Number of modular exponentiations (theoretical comparison)

	Performed by the vote encryption device			Performed by the "election server" and independent third parties		
	Vote encryption		Receipt creation	Vote validity verification	Receipt verification	Canonical vote transf.*
	Candidate encryption	Validity proof*				
MP3	5	5	0	8	5	0
MP1	4.α = 48	-	0	-	2.α = 48	≈ α/2 = 12
MP1a	2 + 4.α = 50	5	0	8 + 2.α =56	2.α = 48	0
MP2	6 + mme	8 + mme	mme	8	8 + mme	3 + mme

α = 24 (commonly proposed value)
mme = 1 matrix modular exponentiation
* MP2 values include our add-ons to complete its specification

Number of modular exponentiations (theoretical comparison)

	Performed by the vote encryption device			Performed by the "election server" and independent third parties		
	Vote encryption		Receipt creation	Vote validity verification	Receipt verification	Canonical vote transf.*
	Candidate encryption	Validity proof*				
MP3	5	5	0	8	5	0
MP1	4.α = 48	-	0	-	2.α = 48	≈ α/2 = 12
MP1a	2 + 4.α = 50	5	0	8 + 2.α =56	2.α = 48	0
MP2	6 + mme	8 + mme	mme	8	8 + mme	3 + mme

α = 24 (commonly proposed value)
mme = 1 matrix modular exponentiation
* MP2 values include our add-ons to complete its specification

Vote Encryption
Implementation results overview

- Vote encryption times for a ballot with 10 candidates

	MP1	MP2	MP3
Number of ciphertexts	480	20	20
JavaCard (|p|=1024, |q|=512)	8.5 min	15 hours	1.5 min
MULTOS card (|p|=1024, |q|=512)	5 min	30 min	43 sec
MULTOS card (|p|=1024, |q|=160)	4 min	2.8 min	28 sec

p and q are cryptosystem parameters (ElGamal parameters)

Conclusions

- MarkPledge 3
  - The more efficient and simpler MarkPledge style voter verifiable encryption.
  - Highly sound and simple voter YES vote verification
  - The only MarkPledge specification that runs in acceptable times on today's constrained hardware (smart cards, secure elements inside smart phones).
  - Can replace the other MarkPledge specifications in previously proposed vote protocols.

Conclusions

technology
from seed

- MarkPledge 3
  - The more efficient and simpler MarkPledge style voter verifiable encryption.
  - Highly sound and simple voter YES vote verification
  - The only MarkPledge specification that runs in acceptable times on today's constrained hardware (smart cards, secure elements inside smart phones).
  - Can replace the other MarkPledge specifications in previously proposed vote protocols.

Questions?