Towards Practical Coercion-Resistant Electronic Elections

Jacques Traoré
France Télécom
Orange Labs R&D

VETO 2009 – 28 June 2009
Grenoble

Outline

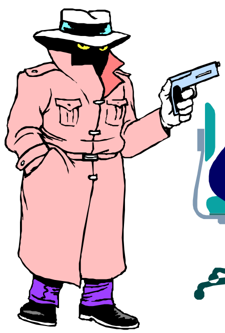
1. Introduction
2. JCJ scheme – a review
3. Our proposal
4. Client/Server trade-offs in universally verifiable elections
5. Conclusion

Electronic Voting Methods

- Supervised voting (off-line voting)

How to guarantee the "What You see is What You Vote For?

- Remote Electronic Voting (on-line voting)




Some desired properties of e-voting systems

- **Eligibility:** only legitimate voters can vote, and only once
- **Universal verifiability:** All voters can verify that the final tally is correct
  - The votes they cast are included
  - Only authorized votes are counted
  - No votes are changed during tallying
- **Privacy:** no adversary can learn any more about votes than is revealed by the final tally
  - Anonymity: hide map from voter to vote
  - Receipt-freeness: prohibit proof of vote
  - *Coercion-freeness*: adaptative

Stronger

Voters cannot prove whether or how they voted, even if they can interact with the adversary while voting.

Basic Tools

- Building Blocks
  - El Gamal cryptosystem (they need a variant of El Gamal in fact for their security proof)

- El Gamal cryptosystem: G a group of prime order p, g a generator of G
  - the secret key is x, the public key is $h = g^{x}$,
  - Encryption of m is $c = (g^{r}, h^{r}m)$,
  - Decryption of c is $(g^{r})^{-x} (h^{r}m)$

El Gamal cryptosystem

- Decryption (private key) can easily be distributed
  - No need to trust a single entity

- Encryption is homomorphic
  - Multiplicative, or additive with a variant: $E_h(m)* E_h(m') = E_h(m*m')$
    - $E_h(m)* E_h(m') = E_h(m*m')$
    - $E_h(m)^k = E_h(m^k)$
  - Computing on encrypted data is easy

- Comparing the plaintexts of two ciphertexts (without decrypting them) is easy:
  - Plaintext Equivalence Test (PET): PET(E_h(m1), E_h(m2) = 1 if m1 = m2 and 0 otherwise

- Re-encryption is easy : mix-nets can be efficiently implemented
  - For simulating an "anonymous channel"
  - For simulating "ballot shuffle"
  - C = (g^r, h^r.m) can be transformed on a new ciphertext C' of *m* without knowing *m* and/or the secret key : C' = ($g^{r+r'}$, $h^{r+r'}$.m)

JCJ scheme* – a review

- Basic ingredients:
  - Voter employs anonymous credential obtained during the registration phase
  - We don’t know who voted (at time of voting) or what was voted
  - Valid credentials are required for vote to count
  - Voter can make "fake credentials" and vote multiple times
  - A coercer cannot tell whether a credential is correct or not
    - Attacker cannot tell whether a vote is valid or not

- Basic idea:
  - To mislead a coercer, the voter sends invalid ballot(s) as long as he is coerced, and a valid ballot as soon as he is not coerced
  - It suffices that the voter finds a window-time during which he is not coerced

* Juels-Catalano-Jakobsson - WPES 2005

Security model

- Registration:
  - Attacker cannot interfere with registration process
- Before voting:
  - Attacker can provide keying or other material to voter (even entire ballot)
- During vote:
  - Votes may be posted anonymously (for strongest security) or semi-anonymously (for weaker guarantees)
  - Bulletin board is universally accessible
- At all times:
  - Attacker has access to all public information, i.e., encrypted and decrypted ballots

Assumption. Voters trust their voting client.

Cast of Characters

Voters

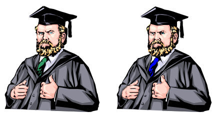
Receive their credential during the registration phase

Registration Authorities

Issue credentials in a distributed manner during the registration phase. They share an El Gamal secret key. R is the corresponding public key

Coercer

Try to verify whether the coerced voter voted as prescribed



Talliers

Manage the tallying process. They share an El Gamal secret key. T is the corresponding public key

Registration

- The authorities generate a random value C_S

Credential List 1

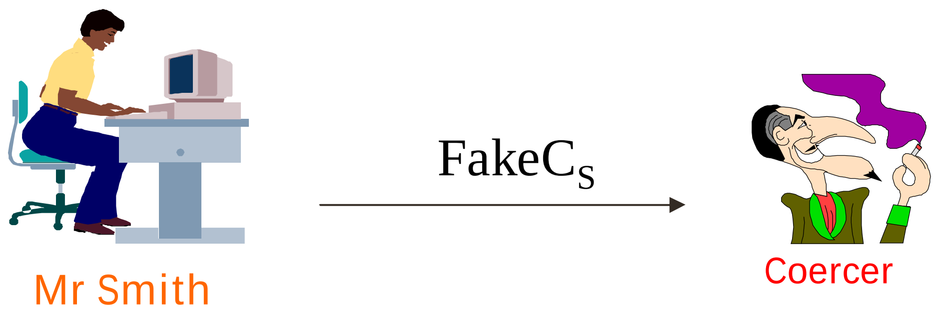
Mr Baker : E_R (C_B)

Mr Durand : E_R (C_D)

Mr Traoré : E_R (C_T)

Mr Smith : E_R (C_S)

- Mr Smith's credential is C_S. He can send a fake credential FakeC_S to the coercer



Voting

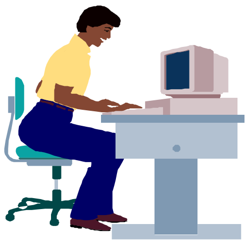
- **Anatomy of a ballot:** (E_T (vote), E_R (Credential), NIZKPs)

  - Vote under coercion

E_T (Bush), E_R ($FakeC_S$), NIZKPs

Bulletin Board 1

  - Revote



Mr Smith

E_T (Gore), E_R (C_S), NIZKPs

Bulletin Board 1

Tallying Ballot

Step 1: Check NIZKPs

Bulletin Board 1

E_T (Bush), E_R (FakeC_B), NIZKP

E_T (Gore), E_R (C_D), NIZKP

E_T (Bush), E_R (FakeC_S), NIZKP

E_T (Gore), E_R (C_S), NIZKP

E_T (Gore), E_R (C_J), NIZKP

~~E_T (Bush), E_R (C_K), NIZKP~~

E_T (Bush), E_R (C_E), NIZKP

.

.

E_T (Bush), E_R (C_J), NIZKP

Tallier 1

Tallier 2

Ballots with invalid NIZKP are discarded

Tallying Ballot

Step 2: Elimination of duplicates using PET

Bulletin Board 2

E_T (Bush), E_R (FakeC_B)
E_T (Gore), E_R (C_D)
E_T (Bush), E_R (FakeC_S)
E_T (Gore), E_R (C_S)
~~E_T (Gore), E_R (C_J)~~ ü
E_T (Bush), E_R (C_E)
.
.
E_T (Bush), E_R (C_J) ü

Authority 1

Authority 2

Keep the last one for example

Tallying Ballot

Step 3: Mixing the ballots

Tallier 1

Tallier 2

Bulletin Board 3
E_T (Bush), E_R ($FakeC_B$)
E_T (Gore), E_R (C_D)
E_T (Bush), E_R ($FakeC_S$)
E_T (Gore), E_R (C_S)
E_T (Bush), E_R (C_E)
.
.
E_T (Bush), E_R (C_J)

Bulletin Board 4
E_T (Bush), E_R (C_J)
E_T (Bush), E_R ($FakeC_B$)
E_T (Gore), E_R (C_S)
E_T (Gore), E_R (C_D)
E_T (Bush), E_R (C_E)
.
.
E_T (Bush), E_R ($FakeC_S$)

Tallying Ballot

Step 4: Mixing the list of valid credentials

Tallier 1

Tallier 2

Credential List 1

Mr Baker : E_R (C_B)

Mr Durand : E_R (C_D)

Mr Traore : E_R (C_T)

Mr Smith : E_R (C_S)

→

→

Credential List 2

E_R (C_D)

E_R (C_S)

E_R (C_B)

E_R (C_T)

Tallying Ballot

Step 5: Checking credentials using PET

Authority 1

Authority 2

Bulletin Board 4
E_T (Bush), E_R (C_J)
~~E_T (Bush), E_R ($FakeC_B$)~~ ü
E_T (Gore), E_R (C_S)
E_T (Gore), E_R (C_D)
E_T (Bush), E_R (C_E)
.
.
~~E_T (Bush), E_R ($FakeC_S$)~~ ü

PET ←→

Credential List 2
E_R (C_D)
E_R (C_S)
E_R (C_B)
E_R (C_T)

Tallying Ballot

Step 6: Decrypt valid votes

Authority 1

Authority 2

Bulletin Board 5

E_T (Bush)

E_T (Gore)

E_T (Gore)

E_T (Bush)

Distributed Decryption

Results

Bush

Gore

Gore

Bush

Drawbacks

- quadratic overhead
  - N the number of voters, V the number of votes ($V \geqq N$)
  - $O(V^2)$ tests for duplicates
  - $O(N^2)$ matching tests
- denial of service attack

Our proposal

- Building Blocks
  - ElGamal Cryptosystem (we also need in fact a variant of El Gamal for our security proof)
  - Mix Net
  - Zero-Knowledge proofs
  - Credentials with a special structure: derived from "Boneh-Boyen-Sacham" or "Camenisch-Lysanskaya" signature schemes (Crypto'04)

Designated verifier *signature* scheme

- Based on "Boneh-Boyen-Sacham's group signature scheme (Crypto 2004)

- Setup:
  - Generators g_0, g, h of a cyclic group G of order p where DDH is hard
  - Public key of the signer: PK = g_0^y,
  - Secret key of the signer: SK = y
- "*Signature*" on a random message *x*:
  - Choose a random value *r*
  - *Compute* $A = (g\, h^x)^{1/(y+r)}$
  - *"Signature" on x = (A, r)*

$$A^y = A^{-r} g h^x \qquad (1)$$
$$A^{y+r} g^{-1} h^{-x} = 1 \qquad (2)$$

- Designated Verification:
  - Prove that $\mathrm{Log}_A(A^{-r}gh^x) = \mathrm{Log}_{g_0}(PK)$ using a *Designated Verifier Proof* (Jakobsson-Sako – Impagliazzo)
  - Only the (designated) verifier can be convinced by this proof

Deciding whether a pair (A, r) is a valid signature on a message x is equivalent to the DDH problem

Cast of Characters

Voters

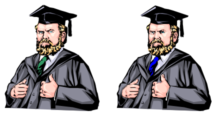
Receive their credential during the registration phase

Registration Authorities

Issue credentials in a distributed manner during the registration phase. They share a secret key of our DVS. R is the corresponding public key

Coercer

Try to verify whether the coerced voter has voted as prescribed



Talliers

Manage the tallying process. They share an El Gamal secret key. T is the corresponding public key

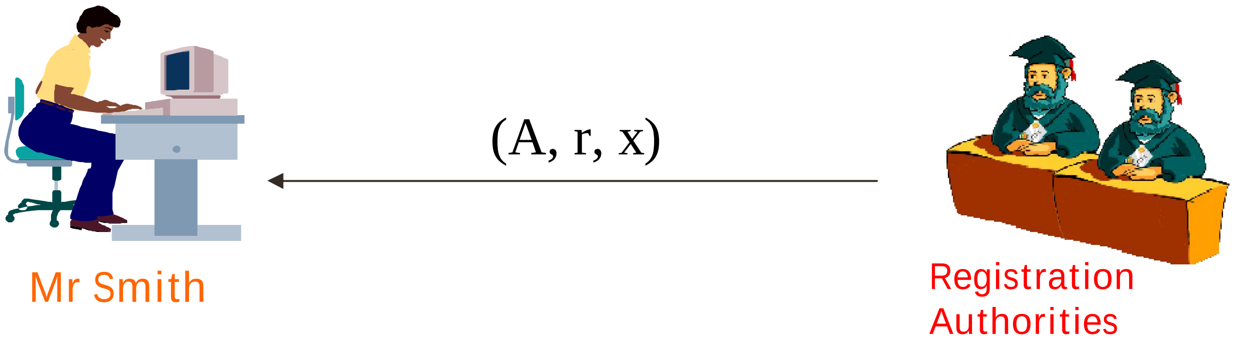
Set-up

- Setup:
  - Generators g_0,g,h,m of a cyclic group G of order p (DDH problem is hard)
  - Registration authority: PK=g_0^y,SK= y
  - Talliers: share y and an ElGamal secret key

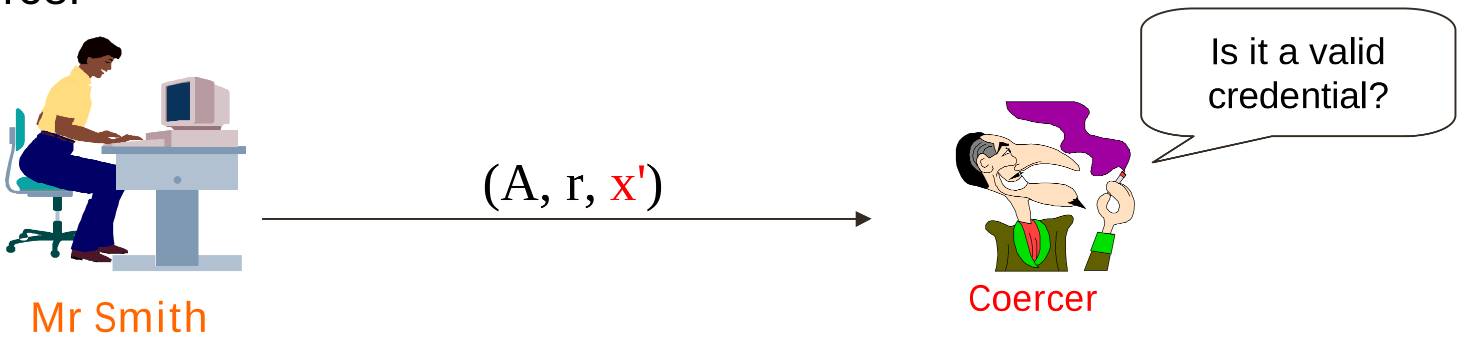
- Registration
  - *Credential: (A,* r, x)
    - *x* and *r* are randomly chosen by R
    - *A* is computed as follows by R: $A = (g\, h^x)^{1/(y+r)}$
  - A credential is valid iff the voter knows two values *x* and *r* such that: A^{y+r}=g h^x (which is equivalent to $A^{y+r} g^{-1} h^{-x} = 1$)
  - Fake credential: (A , r , x')

Registration

- The registration authorities generate in a distributed manner a DVS *signature* (A, r) on a random value *x* and prove to Mr Smith using a DVP that the *signature* is valid



- Mr Smith's credential is (A, r, x). He can send a fake credential (A, r, x') to the coercer



Basic Facts about these credentials

- **A passive coercer can't check if a credential is valid or not under the DDH assumption :** given g, g^a , g^b, g^c decide whether c = ab mod p or not.

- **A coercer can't forge valid credentials under the q-SDH assumption**
  - q-SDH: given g, g^x, …, $g^{(x^q)}$, find a pair (c, A) such that $A^{x+c} = g$

- **An active coercer can't check if a credential is valid or not (under the Strong DDH Inversion (SDDHI) assumption)**

Strong DDH Inversion (SDDHI): Suppose that $g \in \mathbb{G}$ is a random generator of order $q \in \Theta(2^k)$. Let $\mathcal{O}_a(\cdot)$ be an oracle that, on input $z \in \mathbb{Z}_q^*$, outputs $g^{1/(a+z)}$. Then, for all probabilistic polynomial time adversaries $\mathcal{A}^{(\cdot)}$ that do not query the oracle on x,

$$\Pr[a \leftarrow \mathbb{Z}_q^*;\ (x, \alpha) \leftarrow \mathcal{A}^{\mathcal{O}_a}(g, g^a);\ y_0 = g^{1/(a+x)};\ y_1 \leftarrow \mathbb{G};$$
$$b \leftarrow \{0,1\};\ b' \leftarrow \mathcal{A}^{\mathcal{O}_a}(y_b, \alpha) : b = b'] < 1/2 + 1/\text{poly}(k).$$

* The SDDHI assumption holds in generic groups

Anatomy of a ballot

Credential : A tuple (A, r, x) such that $A^{y+r}g^{-1}h^{-x} = 1$

- Ballot
  - $(E_T[vote], E_T[A], E_T[A^r], E_T[h^x], F=m^x, P)$
  - P is a NIZKP of validity, that is :
    - $E_T(vote)$ is an encryption of a valid vote
    - Voter knows the plaintext related to E_T (A)
    - Voter knows the "discrete logarithm" of $E_T[A^r]$ in the base $E_T[A]$
    - Voter knows the plaintext related to $E_T[h^x]$ as well as the discrete logarithm x of this plaintext in the base h.
    - Voter knows the discrete logarithm of F in the base m and that this discrete logarithm is equal to x

Voting

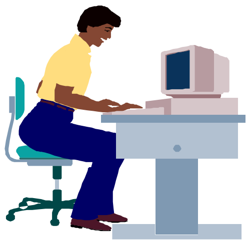
- **Anatomy of a ballot:** $(E_T\ (vote), E_T\ (A), E_T\ (A^r)\ E_T\ (h^x), m^x, Proof)$

  - Vote under coercion

$E_T\ (Bush), E_T\ (A), E_T\ (A^r)\ E_T\ (h^{x'}), m^{x'}, Proof1$ →

Bulletin Board 1

  - Revote



Mr Smith

$E_T\ (Gore), E_T\ (A), E_T\ (A^r)\ E_T\ (h^x), m^x, Proof2$ →

Bulletin Board 1

Tallying phase

Step 1: Discard ballots with invalid proofs

Bulletin Board 1
E_T (Bush), E_T (A), E_T (A^r) E_T ($h^{x'}$), $m^{x'}$, $Proof_1$
E_T (Bush), E_T (B), E_T (B^s) E_T (h^y), m^y, $Proof_2$
E_T (Bush), E_T (C), E_T (C^t) E_T (h^z), m^z, $Proof_3$
E_T (Gore), E_T (B), E_T (B^s) E_T (h^y), m^y, $Proof_4$
~~E_T (Bush), E_T (D), E_T (D^u) E_T (h^v), m^w, $Proof_5$~~
E_T (Gore), E_T (A), E_T (A^r) E_T (h^x), m^x, $Proof_6$

Tallier 1

Tallier 2

Tallying phase

Step 2: Elimination of duplicates: the ballots that have the same fourth component

Bulletin Board 2
E_T (Bush), E_T (A), E_T (A^r) E_T ($h^{x'}$), $m^{x'}$
~~E_T (Bush), E_T (B), E_T (B^s) E_T (h^y), m^y~~
E_T (Bush), E_T (C), E_T (C^t) E_T (h^z), m^z
E_T (Gore), E_T (B), E_T (B^s) E_T (h^y), m^y
E_T (Gore), E_T (A), E_T (A^r) E_T (h^x), m^x

ü

ü

Tallier 1

Tallier 2

Keep the last one for example

Tallying phase

Tallier 1 Tallier 2

Step 3: Mixing the ballots

Bulletin Board 3

E_T (Bush), E_T (A), E_T (A^r) E_T ($h^{x'}$)

E_T (Bush), E_T (C), E_T (C^t) E_T (h^z)

E_T (Gore), E_T (B), E_T (B^s) E_T (h^y)

E_T (Gore), E_T (A), E_T (A^r) E_T (h^x)

Bulletin Board 4

E'_T (Gore), E'_T (A), E'_T (A^r), E'_T (h^x)

E'_T (Gore), E'_T (B), E'_T (B^s), E'_T (h^y)

E'_T (Bush), E'_T (C), E'_T (C^t), E'_T (h^z)

E'_T (Bush), E'_T (A), E'_T (A^r), E'_T ($h^{x'}$)

Reencrypt and permute each row

Tallying phase

Step 4: Checking credentials

Authority 1

Authority 2

Bulletin Board 4

E'_T (Gore), E'_T (A), E'_T (A^r), E'_T (h^x)

E'_T (Gore), E'_T (B), E'_T (B^s), E'_T (h^y)

E'_T (Bush), E'_T (C), E'_T (C^t), E'_T (h^z)

~~E'_T (Bush), E'_T (A), E'_T (A^r), E'_T ($h^{x'}$)~~

1. The authorities compute $C=E'_T\,[A^{y+r}g^{-1}h^{-x}]$ from $E'_T\,[A]$, $E'_T[A^r]$, $E'_T[h^x]$ *and* $SK = y$
2. Test whether C is an encryption of 1
   1. Power C to a fresh random number 'f' and jointly decrypt C^f.
   2. $D[C^f] = 1$? Yes = valid / No = invalid and discard ballots

Tallying phase

Step 5: Decrypt valid votes

Tallier 1

Tallier 2

Bulletin Board 4

E'_T (Gore)

E'_T (Gore)

E'_T (Bush)

Distributed Decryption ↔

Results

Gore

Gore

Bush

Computational Definition of Coercion-Resistance (1)

Experiment $\mathbf{Exp}^{c\text{-}resist}_{\mathsf{ES},\mathcal{A},H}(k_1, k_2, k_3, n_V, n_A, n_C)$

$V \leftarrow \mathcal{A}(\text{voter names}, \text{“control voters”});$

$\{(sk_i, pk_i) \leftarrow \mathsf{register}(SK_{\mathcal{R}}, i, k_2)\}_{i=1}^{n_V};$

$(j, \beta) \leftarrow \mathcal{A}(\{sk_i\}_{i \in V}, \text{“set target voter and vote”});$

if $|V| \neq n_A$ or $j \notin \{1, 2, \ldots, n_V\} - V$ or

$\beta \notin \{1, 2, \ldots, n_C\} \cup \phi$ then

output ‘0’;

$b \in_U \{0, 1\};$

if $b = 0$ then

$\tilde{sk} \leftarrow \mathsf{fakekey}(PK_{\mathcal{T}}, sk_j, pk_j);$

$\mathcal{BB} \Leftarrow \mathsf{vote}(sk_j, PK_{\mathcal{T}}, n_C, \beta, k_2);$

else

$\tilde{sk} \leftarrow sk_j;$

$\mathcal{BB} \Leftarrow \mathsf{vote}(\{sk_i\}_{i \neq j, i \notin V}, PK_{\mathcal{T}}, n_C, D_{n_U, n_C}, k_2)$

$\mathcal{BB} \Leftarrow \mathcal{A}(\tilde{sk}, \mathcal{BB}, \text{“cast ballots”});$

$(\boldsymbol{X}, P) \leftarrow \mathsf{tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3);$

$b' \leftarrow \mathcal{A}(\boldsymbol{X}, P, \text{“guess } b\text{”});$

if $b' = b$ then

output ‘1’;

else

output ‘0’;

The credentials are given to the voters

A sets coercive target

If b = 0 the coerced voter cast a ballot for β and gives a fake credential to A

If b = 1 the coerced voter gives her valid credential to A and does not cast a ballot

A guesses coin flip

$$\mathbf{Succ}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \Pr[\mathbf{Exp}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \text{‘1’}]$$

Computational Definition of Coercion-Resistance (2)

Experiment $\mathbf{Exp}^{c\text{-}resist\text{-}ideal}_{\mathsf{ES},\mathcal{A},H}(k_1, k_2, k_3, n_V, n_A, n_C)$

$V \leftarrow \mathcal{A}'(\text{voter names, "control voters"});$

$\{(sk_i, pk_i) \leftarrow \mathsf{register}(SK_{\mathcal{R}}, i, k_2)\}_{i=1}^{n_V};$

$(j, \beta) \leftarrow \mathcal{A}'(\text{"set target voter and vote"});$

if $|V| \neq n_A$ or $j \notin \{1, 2, \ldots, n_V\} - V$ or

$\beta \notin \{1, 2, \ldots, n_C\} \cup \phi$ then

output '0';

$b \in_U \{0, 1\};$

if $b = 0$ then

$\mathcal{BB} \Leftarrow \mathsf{vote}(sk_j, PK_{\mathcal{T}}, n_C, \beta, k_2);$

$\tilde{sk} \Leftarrow sk_j;$

$\mathcal{BB} \Leftarrow \mathsf{vote}(\{sk_i\}_{i \neq j, i \notin V}, PK_{\mathcal{T}}, n_C, D_{n_U, n_C}, k_2);$

$\mathcal{BB} \Leftarrow \mathcal{A}'(\tilde{sk}, \{sk_i\}_{i \in V}, \text{"cast ballots"});$

$(\boldsymbol{X}, P) \leftarrow \mathsf{ideal\text{-}tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3);$

$b' \leftarrow \mathcal{A}(\boldsymbol{X}, \text{"guess } b\text{"});$

if $b' = b$ then

output '1';

else

output '0';

The credentials are given to the voters

A sets coercive target

the coerced voter evades coercion

A' guesses coin flip but it's only input is the final tally

$$\mathbf{Succ}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \Pr[\mathbf{Exp}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \text{'1'}]$$

Computational Definition of Coercion-Resistance (3)

Definition 1. *We define an election scheme* ES *as* coercion resistant *if for any polynomially-bounded adversary* $\mathcal{A}$, *any parameters* n *and* n_C, *and any probability distribution* D_{n,n_C}, *the quantity*

$$\mathbf{Adv}_{\mathsf{ES},\mathcal{A}}^{c\text{-}resist} = \left|\mathbf{Succ}_{\mathsf{ES},\mathcal{A}}^{c\text{-}resist}(\cdot) - \mathbf{Succ}_{\mathsf{ES},\mathcal{A}}^{c\text{-}resist\text{-}ideal}(\cdot)\right|$$

is negligible in all security parameters for any voter function $\mathcal{V}o$.

Intuitively, this definition means that in a real protocol execution, *A* learns nothing more than the election tally

Our protocol satisfies the coercion-resistant requirement (in the random oracle model) under the SDDHI assumption

Computational Definition of Verifiability

Experiment $\mathbf{Exp}^{ver}_{\mathsf{ES},\mathcal{A}}(k_1, k_2, k_3, n_C, n_V)$

$\{(sk_i, pk_i) \leftarrow \mathsf{register}(SK_{\mathcal{R}}, i, k_2)\}_{i=1}^{n_V}$; % voters are registered

$(\mathcal{BB}, \boldsymbol{X}, P) \leftarrow \mathcal{A}(SK_{\mathcal{T}}, \{(sk_i, pk_i)\}_{i=1}^{n_V}$, "forge election"); % $\mathcal{A}$ concocts full election

$(\boldsymbol{X}', P') \leftarrow \mathsf{tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3)$; % tally is taken on $\mathcal{BB}$

if $\boldsymbol{X} \neq \boldsymbol{X}'$ % does $\mathcal{A}$'s tally differ from correct $\mathcal{BB}$ tally?

and $\mathsf{verify}(PK_{\mathcal{T}}, \mathcal{BB}, n_C, \boldsymbol{X}, P) =$ '1' then % does function verify accept?

output '1';

else

output '0';

$\mathbf{Succ}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \mathsf{Pr}[\mathbf{Exp}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \text{'1'}]$ should be negligible

Our protocol satisfies the verifiability requirement (in the random oracle model) under the q-SDH assumption

Client/Server trade-offs in universally verifiable elections

- Setup:
  - Generators g_0,g,h,m of a cyclic group G of order p (DDH problem is hard)
  - Registration authority: PK=g_0^y,SK= y
  - Talliers: share y and an ElGamal secret key
- Encoding of votes for L candidates:
  - *M*: Upper bound on number of voters.
  - candidate 1 $\rightarrow$ 1, candidate 2 $\rightarrow$ M, . . . , candidate L $\rightarrow$ M^{L-1}.

Generation of the ballots

- Ballot for the candidate j: (A, r, x)
  - *Where* $x = M^j$ and r is randomly chosen by R
  - A is computed as follows by R: $A = (g h^x)^{1/(y+r)}$
- The ballot is valid *iff* : $A^{y+r} = g h^x$ (which is equivalent to $A^{y+r} g^{-1} h^{-x} = 1$)

Voting

- *A vote for candidate j: (E[A], E[A^r], E[h^x], P) where x = M^j*

- *P* is a NIZKP of validity, that is :
  - *E(vote)* is an encryption of a valid vote
  - Voter knows the plaintext related to E(A)
  - Voter knows the "discrete logarithm" of *E[A^r]* in the base *E[A]*
  - Voter knows the plaintext related to *E[h^x]* as well as the discrete logarithm *x* of this plaintext in the base *h*.

Tallying Ballot

Step 1: Discard ballots with invalid proofs

Bulletin Board 1
E_T (A), E_T (A^r) E_T (h^x), $Proof_1$
E_T (A), E_T (A^r) E_T (h^x), $Proof_2$
E_T (C), E_T (C^t) E_T (h^z), $Proof_3$
E_T (D), E_T (D^u) E_T (h^w), $Proof_4$

Tallier 1

Tallier 2

Tallying Ballot

Step 4: Checking valid ballots

Authority 1

Authority 2

Bulletin Board 2

E_T (A), E_T (A^r) E_T (h^x)

E_T (A), E_T (A^r) E_T (h^x)

~~E_T (C), E_T (C^t) E_T (h^z)~~

E_T (D), E_T (D^u) E_T (h^w)

1. The authorities compute $C=E[A^{y+r}g^{-1}h^{-x}]$ from $E[A]$, $E[A^r]$, $E[h^x]$ and $SK = y$
2. Test whether C is an encryption of 1
   1. Power C to a fresh random number 'f' and jointly decrypt C^f.
   2. $D[C^f]$ =1 ? Yes = valid / No = invalid and discard ballots

Tallying Ballot

Step 6: compute the result using the homomorphism

Bulletin Board 3
$E_T\ (h^x)$ $E_T\ (h^x)$ $E_T\ (h^w)$

Tallier 1

Tallier 2

Conclusion

- **The JCJ scheme is promising, but not efficient**
- **We design a practical (with linear work factor), publicly verifiable and coercion- resistant voting scheme (with respect to JCJ's model) for remote elections**
- **Not just practical, but essential for Internet voting!**
- **Open problem: how to remove the assumption related to the voter's computer?**