Coercion Resistance In Voting Systems

Panagiotis Grontas

$\mu\Pi\lambda\forall$ - CoReLab Crypto Group

29/01/2014

Privacy I

Secret Ballot Elections

Privacy < Receipt Freeness < Coercion Resistance

Privacy

A passive adversary cannot guess a vote from the election result and the secret ballots

Receipt Freeness

The voter walks away from voting, unable to costruct a receipt of the actual vote.

The adversary is still passive, but voters might want to sell their votes

What if the adversary is active?

A stronger adversary I

An active attacker might force the voter:

- to vote randomly
- to nullify one vote (vote in the *opposite* manner)
- to abstain from voting
- to give away the secret keys (and be impersonated)

A stronger adversary II

Attack against a mixnet

1. The adversary retrieves the private key of a voter
2. ... selects its own preferred vote v
3. ... Encrypts it with the public key of the mixnet $u = E_(pk_{MN}, v)$
4. ... signs the vote with the private key $s = E(sk_i, E_(pk_{MN}, v))$
5. and appends (u, s) to the BB
6. Signature validates and so the vote is anonymised by the mixnet
7. The adversary can check that the u is in the mixnet, before the anonymisation process begins

or simply ...

1. Skip steps 1-3
2. Give readily a u to the voter

Coercion Resistance: Definition and Requirements I

Coercion Resistance

The adversary cannot tell whether the voter submitted to the coercion attempt

Application: Remote (Internet) Voting

Main Idea [JCJ05]

- Each voter casts many ballots
- Some are cast with invalid credentials
- Some are cast with valid credentials
- Only one vote (that corresponds to a valid credential) counts

Coercion Resistance: Definition and Requirements II

Requirements

- Bulletin Board
- Existence of anonymous channel for vote casting: Lack of anonymous channel enables forced abstention attack.
- Uncertainty about behaviour of some voters (corruption is allowed to some extent)
- Anonymous Voting Credentials
- Ability to create indistinguishable fake credentials
- One moment of privacy (at least)
- Untappable Registration

The JCJ Protocol - High Level Overview

- **Registration** Creation of the voter roll: List of (voter,credential) pairs
- **Voting** Casting of ballots using fake and valid credentials. The system treats them both the same way
- **Tallying**
  - Filter out votes with invalid credentials by comparing them against the voter roll
  - Keep a single vote per voter
  - Count the votes

Cryptographic Primitives

- Randomized Threshold Public Key Crypto that allow for reencryption
- Zero Knowledge Proofs
- Universally Verifiable Mixnets
- Plaintext Equivalence Tests (PET)

Plaintext Equivalence Tests I

PET

A cryptographic primitive to convince a set of participants that two ciphertexts indeed encrypt the same plaintext.
It operates in a distributed setting, where the decryption key is shared among the participants.

Plaintext Equivalence Tests II

An example with El Gamal

- **Input:** Two ciphertexts $(G, M), (G', M')$
- **Output:** True if they encrypt the same plaintext and false otherwise
- **Main idea:** The ciphertext $C_{PET} = (\frac{G}{G'}, \frac{M}{M'})$ will be the encryption of 1 if the ciphertexts encrypt the same plaintext. Otherwise it will decrypt to a random integer.
- Implementation: Each player i
  - Selects a random integer z_i and commits to it using the Pedersen commitment scheme.
  - Blinds C_{PET} using the random value z_i thus producing $(G_i, M_i) = ((\frac{G}{G'})^{z_i}, (\frac{M}{M'})^{z_i})$
  - Everybody proves that the (G_i, M_i) are indeed blinded with the random values.
  - The players calculate $(\prod_i G_i, \prod_i M_i)$ and decrypt by combining their shares of the key.
  - If the decryption yields 1 the test return *true*, otherwise the test return *false*.

The JCJ Protocol - Detailed Description I

1. **Setup Phase:**
    - (pk_R, sk_R) key pairs for registration authorities R
    - (pk_T, sk_T) key pairs for tallying authorities T
    - Minority Corruption for R and T is allowed
2. **Registration Phase:**
    - Validate voter identity
    - Generate the valid voter credentials $(S_i = E_{pk_T}(\sigma_i), \sigma_i)$ where σ_i is random
    - S_i is placed in the BB and σ_i is transmitted through an untappable channel
    - The voter roll: list of the encrypted credentials $VR = \{S_i\}_{i=1}^{N}$
    - This phase must be *corruption-free*:
        - The credential is generated in a distributed fashion
        - The voter must delete all the transcripts from the interaction with the registration authority *or*
        - The coercer cannot corrupt any majority of registration authorities *or*
        - The voters can distinguish the corrupted players.

The JCJ Protocol - Detailed Description II

3. **Voting Phase:**
   - Ballots are cast in the bulletin board as usual.
   - Ballot = (Encryption of candidate choice, Encryption of the credential) - a modified ElGamal Scheme is used
   - $B = (E_{pk_T}(v_i), E_{pk_T}(\sigma_i)) = ((g_1^{r_1}, g_2^{r_1}, v_i h^{r_1}), (g_1^{r_2}, g_2^{r_2}, \sigma_i h^{r_2}))$ where r_1, r_2 are random values
   - Proofs
     - knowledge of vote and credential
     - vote validity, candidate index is valid
     - proof that the same randomness is used
   - Proofs are essential for coercion resistance since:
     - Since the voter roll is public, the attacker might spoof a credential by reencrypting a voter roll entry.
     - An invalid vote might indicate a forced abstention or a randomisation attack.
   - The voting phase takes place using an anonymous channel.
   - **Reasons:** Thwart the forced abstention attack
     make the coercer unaware of the actual vote position in the bulletin board.

The JCJ Protocol - Detailed Description III

4. **Tallying Phase:**
   - Before counting begins, the proofs of correctness are extracted and validated
   - Ballots with invalid proofs are discarded
   - Multiple votes with the same credential are eliminated:
     - Two lists are created, from the rest of the valid ballots L.
     - L_1 contains the encrypted votes
     - L_2 contains the encrypted credentials.
     - Duplicates are removed from L_2 using PET (each encryption in L_2 is compared to all others in L_2)
     - The removal is cascaded to the list with the encrypted votes using a *first vote counts* or *last vote counts* rule.
     - The lists are merged. The result L' contains distinct items and is forwarded to a mixnet.

The JCJ Protocol - Detailed Description IV

- Votes with fake credentials are eliminated:
  - The voter roll is forwarded to the mixnet as well, to be mixed and renecrypted
  - The mixed credential list is compared to the mixed voter roll leading to the discovery of the fake credentials using PET.
  - Both the fake credentials and the corresponding votes are removed
- The resulting list of encrypted votes is threshold-decrypted and the votes are counted.

Proof of security I

Coercion game

- The target voter flips a coin b.
- If $b = 0$ then the voter evades coercion by using a fake credential.
- If $b = 1$ the voter submits to coercion.
- In both cases the adversary dictates the selection
- Coercion resistance is achieved when the adversary has a negligible advantage on guessing b

Proof of security II

Proof Strategy

- Target: $\mathcal{A}$ has negligible advantage compared to $\mathcal{A}'$ that is passive, and interacts with an ideal system that (disregards corrupted voters and invalid ballots)
- Strategy: Replace election scheme with simulated scheme. If $\mathcal{A}$ cannot distinguish the two, then $\mathcal{A}$ does not learn anything from the scheme
- $\mathcal{A}$ is equivalent to $\mathcal{A}'$

Proof of security III

The real experiment $c - \textit{resist}$

- The adversary $\mathcal{A}$ selects the targets for corruption.
- Voters register
- $\mathcal{A}$ sets coercive target and choice
- The voter flips a coin b
- If $b = 0$ then the voter makes a voting choice and returns a fake credential to the adversary
- If $b = 1$ then the voter votes, and returns the real credential to the adversary
- $\mathcal{A}$ posts to the bulletin board the votes of the corrupted voters
- The tallying authorities count the votes
- $\mathcal{A}$ gets the result and tries to guess if the coercion attempt succeeded, by setting b'
- If $b' = b$ then output 1 else output 0

Proof of security IV

The ideal experiment $c - resist - ideal$**:Only information is the tally result**

- The adversary $\mathcal{A}'$ selects the targets for corruption.
- Voters register
- $\mathcal{A}$ sets coercive target and choice
- The voter flips a coin b
- If $b = 0$ then the voter makes a voting choice
- The voter return the real credential to the coercer
- $\mathcal{A}'$ posts to the bulletin board the votes of the corrupted voters

Proof of security V

- The tallying authorities count the votes in an ideal way, so that $\mathcal{A}'$ does not learn anything from the tallying
    - Ballots by honest voters are counted normally
    - Double votes are not counted
    - Corrupted players' votes are not counted
    - If $b = 0$ (evade corruption) the vote is not counted
    - If $b = 1$ (submit to corruption) the vote is counted
- $\mathcal{A}'$ gets the result and tries to guess if the coercion attempt succeeded, by setting b'
- If $b' = b$ then output 1 else output 0

Proof of coercion resistance

Theorem

If the Decisional Diffie-Hellman (DDH) Assumption holds, then the protocol provides coercion resistance

Proof.

- Construct a simulator $\mathcal{S}$
- $\mathcal{S}$ takes the honest votes and simulates $c-resist$
- Instead of the votes $\mathcal{S}$ provides random ciphertexts
- Due to the DDH the adversary cannot distinguish the simulated ciphertexts with the real ciphertexts that would come up in an election

$\square$

Discussion I

Drawbacks

- v is the number of **votes** and N is the number of voters
- $v >> N$ because of the fake and multiple votes
- *Complexity*: $O(v^2) + O(vN)$ for duplicate and fake removal
- *Application*: Fake and Real Credential Management

Cut down performance to linear time

- A standard solution: Replace pairwise PET comparisons with hashtables
- Add structure to the credentials

Discussion II

Practical Aspects: Representation and usability of credentials

- Use only two credentials [Sch06]
  - The real one and the fake one, indistinguishable in form and content
  - The property that makes a credential valid is that it is included in the voter roll.
  - The two credentials are included in an observer: a tamper resistant device like a smartcard.
- Panic Passwords

Discussion III

Basic Attacks [Smi05]

The 1009 Attack

- The coercer forces the voter to vote a particular number of times (eg. 1009) using a particular ciphertext
- Check if 1008 votes were removed from L, during deduplication
- Check that the remaining ciphertext was present in the final list

The timestamping attack

- The timestamp of the ballot can serve as a ballot identifier, - receipt
- It must be removed from all outputs

Board Flooding Take advantage of large processing time and create a DoS attack by injecting random votes with invalid credentials

Hashtables instead of PET

An obstacle

A hashtable cannot be used to identify identical credentials because they are subject to randomised encryption

The randomisation of the encryption must be removed before the hashtable is employed

The encrypted credentials must be blinded

Process

- $(u, v) = (g^r, \sigma h^r)$ is an El Gamal ciphertext of credential σ
- s a shared private key, $h = g^s$ the public key, r the randomness employed
  - The authorities select a second private key z which is shared as well.
  - They blind the ciphertext by computing $(u^{sz}, v^z) = (g^{rsz}, \sigma^z g^{rsz})$
  - Subsequently they divide the components which leaves only the plaintext credential blinded σ^z
  - The hashing approach can now be used to half [Smi05] or to the whole [WAB07] of σ^z

Attacks

A variant of the tagging attack AraujoFT07

- The coercer should be able to retrieve the encryption of the voter's credential
- Utilise the malleability of the encryption scheme and construct a ciphertext that is somehow related to the original
- The coercer can compute the product of the encrypted credential and the encryption of a random value known to him
- Discover if the particular vote passed the credential validity test by checking if it was removed from the final output

Does not apply to duplicate removal phase!

Civitas: An implementation I

To generate a credential, a set of registration tellers cooperate:

- Each teller generates a share s_j of the credential, which is encrypted using standard ElGamal yielding S_j, which is placed in the bulletin board.
- The public part of the credential is calculated as $S = \prod_j S_j$
- Due to the multiplicative homomorphism of El Gamal $S = Enc(\prod_j s_j)$

The untappable channel is implemented using the concept of designated verifier proofs through an untappable channel.

- Each teller gives its credential share to the voter and a mechanism to validate it.
- More specifically the teller presents $(s_j, r, S_j^{'} = Enc(s_j, r), D)$ where D is a DVP that $S_j^{'}$ is a reencryption of S_i
- The voter is convinced of the share validity, but cannot transfer the proof to the coercer

Civitas: An implementation II

- The full private credential is constructed by multiplying received shares.

In order to fake the credential, a coerced voter can simply select a fake credential share $\hat{s_j}$, which is assumed belongs to a teller not controlled by the adversarys. Its existence is an assumption of both [CCM08] and [JCJ05].
To solve the scalability problem, Civitas employs parallelism.

- The voters are partitioned in *virtual precincts*, called blocks.
- Vote authorisation and tallying happens indepently in each block, and the results are aggregated.
- As a result the complexity of the scheme is $O(BM^2) + O(BKM)$ where M is the maximum number of votes per block, B is the number of blocks and K is the minimum number of voters per block.

Efficiency due to extra information [SKHS12] I

- Registration Phase: The voter retrieves through the untappable channel the index where his credential is located in the voter roll
- He encrypts and includes the index in the cast vote.
- Duplicate removal through hashtables
- During the tallying phase the index will be decrypted and the credential exactly pinpointed in the voter roll
- A single PET can identify the fake credential

Efficiency due to extra information [SKHS12] II

An observation

- Invalid vote: invalid credential or invalid index
- Invalid index: 2 cases
  - Index in range but not the one in the voter roll: will be revealed in the end
  - Index out of range: will be revealed in the beginning

Fake Indices should be uniform

Fake Indices Provided by the registration authorities

Anonymity Sets - Selections [CH11] I

Anonymity Sets

A set of *identical items* one of which is relevant the rest being decoys.

Application: Include in the ballot the real and $\beta - 1$ other credentials

- Encrypt credential σ using exponential El Gamal
- Commit to g^{σ} and rerandomise public counterpart
- Pick $\beta - 1$ public parts and embed them in the ballot
- Use a WID proof to prove that the credential is a rerandomisation of one of the credentials, without revealing which
- Duplicates are instantly revealed by g^{σ}
- Real votes are deduced by a single PET

A variation [SKHS12] anonymity set is selected by the voter and the authorities create the ballots

Adding structure to the credentials [AFT07] I

- Credentials: A quadruple $(r, \alpha, b = \alpha^y, c = \alpha^{x+xry})$ where:
  - x, y are two secret keys
  - α is a random number $\alpha \neq 1$
  - r is a random number from a group with hard DDH
- A adversary cannot tell whether a quadruple is a valid one or it merely consists of randomly selected components
- An attacker that comes across quadruples $(r_i, \alpha_i, b_i, c_i)$ cannot use them to create a distinct valid one (the LRSW assumption - Lysyanskaya, Rivest, Sahai, and Wolf (LRSW))
- The r part should be kept secret, while (α, b, c) could be made public

Adding structure to the credentials [AFT07] II

Operation

- **Setup**
  - Two generators g, o of a group of prime order where the DH assumption holds.
  - g will be used for the standard ElGamal operations
  - o will be used for duplicate removal in linear time.
  - Registration Authorities Keys $(x, R_x)(y, R_y)$
  - Tallying Authorities Keys $\hat{T}, T$
- **Registration**
  - Validate identity
  - Credential and DV proof transmission through untappable channel
  - **Note:** If the credential (r, α, b, c) is valid, then every credential of the form (r, α^l, b^l, c^l) is valid.
  - **Application:** Many Elections with a single credential
- **Voting**

Adding structure to the credentials [AFT07] III

- Ballot: $(E_T[vc], E_T[\alpha^r], E_T[b^r], E_T[c], \alpha, o^r)$
- Proofs:
  - vc is a valid candidate selection
  - proofs of knowledge of each encrypted plaintext
  - proof that the blinding factor r is the same in $E_T[\alpha^r]$ and o^r

Adding structure to the credentials [AFT07] IV

- **Tallying**
  - **Proof Validity Check** The proofs of the ballots are checked and the ones which are not verified are eliminated
  - **Duplicate Removal** The value o^r is passed through a hashtable and based on it the duplicate ballots are removed.
  - Afterwards for all unique ballots the proofs as well as the value o^r are removed. The plaintext α is encrypted using the T.
  - **Anonymisation** The unique ballots are anonymised via a verifiable reencryption mixnet. The output of the mixnet is a quintuple $(t, u, v, w, z) = (Reenc(vc), Reenc(\alpha), Reenc(\alpha^r), Reenc(b^r), Reenc(c))$
  - **Credential Validity Check** The identification of the invalid ballots is done with the cooperation of the registration authorities which is an active participant in the whole process. Instead of supplying the voter roll as in [JCJ05], they supply their private keys (x, y) in order to:
    - Compute $v^y = Reenc(\alpha^r)^y = Reenc(\alpha^{ry})$
    - Check if $PET(v^y, w) = 1$.

Adding structure to the credentials [AFT07] V

  - If everything is OK the check should be successful since $v^y = Reenc(\alpha^{ry})$ and $w = Reenc(b^r) = Reenc(\alpha^{y^r})$, which means that both are encryptions of α^{ry}
  - Compute a new shared key k to blind $(\frac{z}{(uw)^x})^k = (\frac{Reenc(c)}{(Reenc(\alpha)^x Reenc(b^r)^x)})^k$.
  - If the credential is valid then the result of the computation should decrypt to 1.
  - **Why blinding** If the credential is invalid then $\frac{Reenc(c)}{(Reenc(\alpha)^x Reenc(b^r)^x)}$ might yield information about the relationship between α, b, c in order to construct a fake but valid credential

- **Result Computation** The valid ballots are decrypted and aggregated.

Remarks

- Linear time,deduplication with hashtables on o^r
- Problem with revocation: split the credentials to public (α, b, c) and private r
- Authorities can generate rogue valid credentials

A practical aspect I

Question

How does the voter actually create the fake credentials when under coercion?
Solution: *panic passwords* [CH] as used in the *Selections* scheme ([CH11])

- During registration the voter selects a password to be used during vote casting.
- In reality the voter partitions the space of possible passwords into three categories:
  - The actual password to authenticate the voter and submit the real credential
  - The panic passwords which indicate that the user is under coercion and generate fake credentials. The interaction however is indistinguishable from the one that takes place with the actual password.
  - The inadmissible passwords that indicate authentication failure and do not allow vote casting

A practical aspect II

- Panic passwords should be a sparse subset of inadmissible passwords
- The actual password is a pre selected panic password.
- An example implementation *5-Dictionary*.
  - The password space consists of any combination of five words.
  - The valid passwords are any combination of five words from an agreed upon dictionary.
  - A particular combination is selected as the actual password during registration.
  - Any other combination from the dictionary is a panic password.
  - Any other five word combination from the password space is invalid.

Board Flooding

A DoS attack

Inject fake votes to cause a quadratic effect in the vote authorisation phase

A solution: Dummy credentials and a *smarter* bulletin board

- Extra anonymous tokens are given during registration
- They aim to put an upper bound on the number of ballots
- The bulletin board immediately rejects duplicate ballots and fake ballots, using the linear hashtable based approach (The tagging attack does not apply since the attacker cannot tell do not reach the tallying phase)
- A vote is counted only if it does not correspond to the dummy credential
- The number d of dummy credentials per voter is crucial
  - If it is fixed then the scheme is not coercion resistant (the attacker simply forces the voter to cast $d+1$ ballots
  - A variable number d_i still leaves some voters exposed
    - The coercer might force the voter to vote $min\{d_i\}+1$ times or $max\{d_i\}+1$ times

References I

Roberto Araújo, Sébastien Foulle, and Jacques Traoré.
A practical and secure coercion-resistant scheme for remote elections.
In *Frontiers of Electronic Voting*, 2007.

Roberto Araújo, Narjes Ben Rajeb, Riadh Robbana, Jacques Traoré, and Souheib Yousfi.
Towards practical and secure coercion-resistant electronic elections.
In *CANS*, pages 278–297, 2010.

Roberto Araújo and Jacques Traoré.
A practical coercion resistant voting scheme revisited.
In *VOTE-ID*, pages 193–209, 2013.

References II

Michael R. Clarkson, Stephen Chong, and Andrew C. Myers.
Civitas: Toward a secure voting system.
In *IEEE Symposium on Security and Privacy*, pages 354–368. IEEE Computer Society, 2008.

Jeremy Clark and Urs Hengartner.
Panic passwords authenticating under duress.

Jeremy Clark and Urs Hengartner.
Selections: Internet voting with over-the-shoulder coercion-resistance.
IACR Cryptology ePrint Archive, 2011:166, 2011.

Ari Juels, Dario Catalano, and Markus Jakobsson.
Coercion-resistant electronic elections.
In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 61–70. ACM, 2005.

References III

Reto E. Koenig, Rolf Haenni, and Stephan Fischli.
Preventing board flooding attacks in coercion-resistant electronic voting schemes.
In *SEC*, pages 116–127, 2011.

Jorn Schweisgut.
Coercion-resistant electronic elections with observer.
In Robert Krimmer, editor, *Electronic Voting*, volume 86 of *LNI*, pages 171–177. GI, 2006.

Michael Schlapfer, Rolf Haenni, Reto E. Koenig, and Oliver Spycher.
Efficient vote authorization in coercion-resistant internet voting.
In Aggelos Kiayias and Helger Lipmaa, editors, *VOTE-ID*, volume 7187 of *Lecture Notes in Computer Science*, pages 71–88. Springer, 2011.

References IV

Oliver Spycher, Reto Koenig, Rolf Haenni, and Michael Schlapfer.

A new approach towards coercion-resistant remote e-voting in linear time.

In *Proceedings of the 15th international conference on Financial Cryptography and Data Security*, FC'11, pages 182–189, Berlin, Heidelberg, 2012. Springer-Verlag.

Warren D. Smith.

New cryptographic voting scheme with best-known theoretical properties, June 2005.

Stefan G. Weber, Roberto Araujo, and Johannes Buchmann.

On coercion-resistant electronic elections with linear work.

In *ARES*, pages 908–916. IEEE Computer Society, 2007.