Receipt Freeness In Voting Systems

Panagiotis Grontas

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

06/12/2013

Context

- **Voting Privacy**: Leak no information about voting, to enable secret ballot elections
- Privacy: During and after voting
- **Voting Verifiability**: Leak enough information to enable everybody to check the result
- **Facts**:
  - Voters might want to reveal their votes after the elections
  - Verification information might aid them to
- Privacy should be enforced
- Privacy and verifiability are contradicting
- Verifiability without usable evidence

Receipt Freeness [BT94] I

Receipt-Freeness

The voter cannot prove how he voted after the fact

The protocol must maintain verifiability

- In traditional voting schemes the voter is isolated during the voting process
- He brings nothing outside of the voting booth
- An electronic analogue is needed

Definition

A voting booth is a component that provides information theoretically secure communication between two entities.

- Voting is done in *booth mode*
- The voter can forge all messages he receives.

Receipt Freeness [BT94] II

- It is not enough

E-Voting Schemes always yield a receipt

- Public Key Encryption is randomised to be secure
- The randomisation can server as a receipt
- A coercer might use it to check if the voter cooperated or not
- Simply encrypt the selection using the randomness and check the BB for the ciphertext

Receipt Freeness [BT94] III

Main idea

- Adaptation of the first verifiable voting scheme [CF85],[BY86]
- The voter does not construct the ballot
- Instead, it is constructed by the authorities
- The voter simply selects his choice, without providing any input
- Every action is proved

Receipt Freeness [BT94] IV

1. **The authority** generates $\eta + 1$ ballots as randomised encryptions of a $yes - vote$ and a $no - vote$ in random order.
2. In *booth mode*, the voter receives the decryption of a specific component for each of the $\eta + 1$ ballots.
3. The authority proves publicly that the ballots are of the correct form as in [CF85].
   - The beacon generates η random bits c_i
   - $c_i = 0$: the current ballot is decrypted with a proof of correctness
   - $c_i = 1$: Prove that the master ballot contains valid encryptions by connecting the current ballot with the master ballot
4. The voter *publicly indicates* the preferred index in the first ballot, to be his vote.

Receipt Freeness [BT94] V

- Each authority A_j selects a blinding factor f_j and supplies its encryption e_j to each voter
- Inside the voting booth, the voter is given f_j and proof that its encryption is e_j (using the beacon)
- The voter selects the vote v_i and a polynomial P_i in order to share the votes to the authorities. $P_i(0) = v_i$
- Each share is blinded by calculating $y_j = P(j) - f_j$
- Each voter releases the pair (y_j, e_j) for all authorities. This tuple is the actual vote
- The authorities add the blinded shares and multiply the encryptions for all the voters. Since the encryption is homomorphic the product of the e_j can be decrypted and used to unblind the sum of the blinded shares
- Each authority now has a point for the combined voter polynomial $Q = \sum_{i=1}^{N} P_i$ where $Q(0) = \sum_{i=1}^{N} P_i(0)$ yields the election result
- At least t authorities can reconstruct the polynomial and retrieve the tally

Receipt Freeness [BT94] VI

[BT94] Receipt Freeness was rebutted in [HS00]

- Turn the validity argument around
- Commit to an ordering of the test ballots $(v_0, v_1, ..., v_\eta)$
- The receipt is the commitment
- Each beacon bit validates the receipt
- Probability of receipt validity = Probability of vote validity $= 1 - 2^{-\eta}$

Adding Receipt Freeness to Mixnets [KS95] I

Definition

Chameleon Bit Commitments (Chaum, Brassard and Crepeau) A bit commitment that can open in both 0 and 1 using a trapdoor

Definition

An untappable channel is an addon to the voter that provides information theoretically secure one way communication to another party.

Requirement

The channels used by the decommitments are *physically untappable*.

- [KS95]: the first verifiable mixnet
- ...providing receipt freeness
- The encrypted votes are not be prepared by the voter, but by the system

Adding Receipt Freeness to Mixnets [KS95] II

- The system must convince the voter of the validity of the encryption in a way that is not transferable to a coercer.
- [KS95] utilises the mixnet *backwards*
  - The last mix server prepares encryption of a pair of *yes/no* votes in random internal order for each voter.
  - He shuffles and reneencrypts the lists and proves the correctness of the action using the cut and choose protocol.
  - He then sends the list to the next (previous) mix server, which is actually the previous one
  - Each mix server rearranges randomly the internal order of the votes and proceeds to his normal reencryption and permutation

Adding Receipt Freeness to Mixnets [KS95] III

- The voter must pinpoint the encryption destined for him in the shuffled list and choose internally his preferred option, either yes or no.
- He must know the internal reorderings, the shuffles and the reencryptions.
- Each mix server must commit to these actions and send the commitments to the voter
- The opening of the commitments will provide the necessary information to the voter.
- It must be done however in a manner that does not provide a receipt - *chameleon blobs*
- In order to vote, the voter selects the correct component and uses the mixnet in the normal forward fashion.

Designated Verifier Proofs [JSI96] I

- ZK proofs offer public verifiability
- … The *coercer* can verify too …
- The proof can serve as a receipt
- Ability to forge a proof is needed
- Proofs verifiable only by a designated verifier

Designated Verifier Proofs [JSI96] II

Idea

- The prover wants to prove statement Φ to the verifier
- Construct *verifier-specific version* $\Phi \bigvee DV$ where *DV=I am the designated verifier*
- *DV* corresponds to knowledge or possession of some secret trapdoor information on a chameleon commitment scheme
- Only the designated verifier has access to the trapdoor information (a secret key).
- If the proof is valid then Φ is valid
- Transfer the proof: *DV* is true so the disjunction is always true

Designated Verifier Proofs [JSI96] III

Implementation using Pedersen commitment

- Let $x_V, y_V = g^{x_v}$ be a secret public key pair.
- The prover commits to w by computing $g^w y_V^r$
- To open the commitment the prover sends (w, r)
- The verifier can validate the commitment since he knows x_V
- The verifier can present to an adversary (w', r') such that the commitment is valid

Adding Receipt Freeness to Homomorphic Schemes [HS00] I

Idea

- Combine [KS95] with [CGS97]
- Free the voter from the need to encrypt his choice
- The voting authorities prepare encryptions of the possible votes and the voter simply picks the choice of interest
- Maintain ballot secrecy using a re-encryption mixnet.
- Notify voter of all the permutations and reenecryption factors used, to enable him to pinpoint the vote of choice
- Use of designated verifier proofs
- Requirement: *untappable one way communication channels* from the authorities to the voters

Adding Receipt Freeness to Homomorphic Schemes [HS00] II

- **Vote Generation**
  1. Each valid choice $v_i \in \{v_1, v_2, \cdots v_C\}$ is deterministically encrypted by the public key encryption algorithm (using predetermined randomness) producing the list of the encryptions of the valid choices $L_0 = \{e_1^0, \cdots, e_C^0\}$. Everybody can validate these encryptions by encrypting each choice with the randomness agreed.
  2. Each authority A_k reencrypts and permutes L_{k-1} thus producing L_k
  3. Each authority *publicly* proves that each vote in L_{k-1} has a reencryption in L_k.
  4. Each authority sends using the untappable channel the permutation π_k it applied and *privately* proves that it is valid.
- **Vote Casting** Using the permutation information received from each authority the voter tracks down the output item that encrypts his choice. The vote casted is the index of the desired item.
- **Vote Tallying** Since the scheme applies to homomorphic cryptosystems, the authorities compute the homomorphic function and decrypt the tally, while proving its correctness.

Adding Receipt Freeness to Homomorphic Schemes [HS00] III

WID Proof of reencryption *(for verifiability)*

Input An encrypted vote (x, y) and a list of encrypted votes $L = \{(x_i, y_i)\}_{i=1}^{L}$

Output Proof that there exists a reencryption of (x, y) in L, without revealing the position

Private Input for prover t the index of the reencryption $(x_t, y_t) = (g^{\xi}x, h^{\xi}y)$

- The prover randomly selects $\{d_i\}_{i=1}^{L}, \{r_i\}_{i=1}^{L}$
- He calculates $\{\alpha_i = (\frac{x_i}{x})^{d_i} g^{r_i}\}_{i=1}^{C}$ and $\{b_i = (\frac{y_i}{y})^{d_i} h^{r_i}\}_{i=1}^{C}$.
- For the actual reencrypted value the prover calculates $\alpha_t = g^{\xi d_t + r_t}$ and $b_t = h^{\xi d_t + r_t}$.
- The prover offers the calculated values to the verifier.
- The verifier randomly selects a challenge c and returns it to the prover.

Adding Receipt Freeness to Homomorphic Schemes [HS00] IV

- The prover recalculates d_t as $d'_t = c - \sum_{i=1, i \neq t}^{C} d_i$ and finds r'_t such that $\xi d_t + r_t = \xi d'_t + r'_t$.
- Then he updates the values of d_t, r_t to d'_t, r'_t.
- Finally he submits the values $\{d_i\}_{i=1}^{C}, \{r_i\}_{i=1}^{C}$ to the verifier.
- The verifier validates that:
  - $c = \sum_i d_i$
  - $\{\alpha_i = \frac{x_i}{x}^{d_i} g^{r_i}\}_{i=1}^{C}$
  - $\{b_i = \frac{y_i}{y}^{d_i} h^{r_i}\}_{i=1}^{C}$.

Adding Receipt Freeness to Homomorphic Schemes [HS00] V

DV Proof of reencryption

Peceipt Freeness

Prove that (x', y') is a reencryption of (x, y) with witness ξ
Equivalently $(x', y') = (g^{\xi}x, h^{\xi}y)$
The verifier has a (public,secret) key pair (h_{ver}, z_{ver}).
Use the private key in order to forge the proof for the coercer, while maintaining its validity.

- The prover randomly selects d, w, r and sends to the verifier the values
  - $\alpha = g^d$
  - $b = h^d$
  - $s = g^w h_{ver}^r = g^{w + z_{ver} r}$
- The verifier selects a random challenge c and sends it to the prover.
- The prover calculates $u = d + \xi(c + w)$ and sends u, w, r to the verifier.

Adding Receipt Freeness to Homomorphic Schemes [HS00] VI

- The verifier validates that:
  - $s = g^w h_{ver}^r$
  - $g^u = \frac{x'}{x}^{c+w} \alpha$
  - $h^u = \frac{y'}{y}^{c+w} b$

How to forge the proof

- The verifier knows the secret key
- Find w', r' such that $w + z_{ver} r = w' + z_{ver} r'$
- Forge the proof for any (x'', y''), by randomly selecting the offers α, b and calculating $w' = \alpha - c$ and $r' = \frac{b-w}{z_{ver}}$

References I

Josh Benaloh and Dwight Tuinstra.

Receipt-free secret-ballot elections (extended abstract).

In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, STOC '94, pages 544–553, New York, NY, USA, 1994. ACM.

Josh C Benaloh and Moti Yung.

Distributing the power of a government to enhance the privacy of voters.

In *Proceedings of the fifth annual ACM symposium on Principles of distributed computing*, PODC '86, pages 52–62, New York, NY, USA, 1986. ACM.

Josh D. Cohen and Michael J. Fischer.

A robust and verifiable cryptographically secure election scheme (extended abstract).

In *FOCS*, pages 372–382, 1985.

References II

Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers.

A secure and optimally efficient multi-authority election scheme.

pages 103–118. Springer-Verlag, 1997.

Martin Hirt and Kazue Sako.

Efficient receipt-free voting based on homomorphic encryption.

In *Proceedings of the 19th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'00, pages 539–556, Berlin, Heidelberg, 2000. Springer-Verlag.

Hugo Jonker, Sjouke Mauw, and Jun Pang.

Privacy and verifiability in voting systems: Methods, developments and trends.

Cryptology ePrint Archive, Report 2013/615, 2013.

http://eprint.iacr.org/.

References III

Markus Jakobsson, Kazue Sako, and Russell Impagliazzo.

Designated verifier proofs and their applications.

In Ueli M. Maurer, editor, *EUROCRYPT*, volume 1070 of *Lecture Notes in Computer Science*, pages 143–154. Springer, 1996.

Joe Kilian and Kazue Sako.

Receipt-free MIX-type voting scheme - a practical solution to the implementation of a voting booth.

In *Proceedings of EUROCRYPT 1995*. Springer-Verlag, 1995.