Contribution Title*

No Author Given

No Institute Given

Abstract. Receipt-freeness and coercion-resistance are essential building blocks in the design of privacy-preserving remote electronic voting systems. Our approach enhances the vote privacy, in such a way that neither the voting authority nor a third party can link the decrypted ballot to a particular voter. In particular, this property can be enhanced to the vote privacy violations by the adversaries with high computational power in the future such as quantum computers. We propose a verifiable receipt-freeness based on distributed voting servers, zero-knowledge proofs, and mix-networks.

Keywords: Verifiable elections · Receipt-freeness · Vote privacy.

1 Introduction

Remote electronic voting that allows voters to cast their ballots from their own devices is vulnerable to two major threats: *vote buying* and *voter coercion*. An electronic voting is *receipt-freeness* when voters are unable to convince a third party of their votes even if they want to do so. This property prevents vote buying and is stronger than *ballot privacy*. However, ballot privacy only ensures the privacy of honest voter's vote. Therefore, designing voting protocols that can achieve receipt-freeness alongside *verifiability* is challenging. Verifiability property guarantees that a voter can verify her ballot in the ballot box; the ballot box contains the ballots of eligible voters and the consistency of the counting result with the ballots in the ballot box.

Voting protocols that achieve receipt-freeness simultaneously with verifiability require one or more of the following assumptions: intense interaction between the election authorities and voters, an untappable channel, anti-coercion strategies such as fake credential or re-voting, trust assumptions on voting servers, and costly tallying process.

The strong form of the receipt-freeness scheme is called coercion-resistant. The coercion-resistant property prevents two major attacks: the first is preventing an attacker from obtaining a voter's secret key after the registration phase. The second is a forced-abstention attack that refrains a voter from voting. Juels, Catalano, and Jakobsson proposed a verifiable coercion-resistance scheme [18] that is implemented as a Civitas [12]. It requires untappable channels during the registration, anonymous channels for casting a vote, and has quadratic-time

* Supported by organization x.

complexity based on the number of ballots in the bullet box in the tally phase. However, it provides less interaction between the voter and authorities or ballot box compared to [24, 11]. The receipt-freeness property of KTV-Helios [20, 4] and BeleniosRF [9] is relied on the trust assumption of a trustee and a voting server. The former adds dummy ballots, and the latter re-randomizes the ballots. Also, in these schemes, it is assumed that the registrar is trusted.

In most of the above schemes, vote privacy depends on computational assumptions and is therefore susceptible to adversaries with high computing power, such as quantum computers. An approach to address this challenge is *everlasting privacy*. That is, designing e-voting schemes that remove the link between the vote and the voter apart from computational assumptions. Traditional voting booths, untappable channels, or trust in authorities can offer everlasting privacy. This paper focuses on privacy property towards an adversary with high computing power, receipt-freeness, and verifiability with a reduced trust assumption.

My note: The main idea behind our scheme is that a voter's identity remains unlinkable to her vote during the election process, even for the election authorities. The voter's identity is no longer based on the signature or credential generated and issued by the registrar. A zero-knowledge proof ensures whether a ballot is cast by an eligible voter in the voting phase. The voter first commits to her key in the registration phase. During the voting phase, the voter proves the connection of her voting key to the eligible voter list without revealing her identity. The registrar is semi-trusted, namely in charge of honestly publishing the registered voters. The registrar neither can vote nor alter the ballot posed by the voters or leak the voter credential as in [4, 9, 18].

1.1 Contribution

A receipt to vote in the remote e-voting systems allows a voter to verify that her ballot has been cast, but it also reveals the voter's vote to third parties. As a result, such voting systems are vulnerable to vote buying and coercion. Receipt-freeness is therefore an imperative aspect of vote privacy when the voter is dishonest.

Our protocol provide the following properties:

- *Post-quantum Privacy.* A voter commits to her secret using a collision-resistance hash commitment scheme. The voting serial number and all the information contained in a ballot are not linkable to the voter's identity and the corresponding commitment. Hash functions are secure against quantum computers by increasing the key size. By using a perfectly hiding commitment scheme, our scheme provides everlasting privacy. Therefore, a powerful adversary, such as quantum computers that can break the encryption scheme, would not be able to identify a voter from a ballot and public data.
- *Receipt-freeness.* The receipt-freeness is achieved by using the distributed voting servers that hide the number of votes cast by an individual voter. Although a voter can hand over her secret (the commitment and randomness) after voting, the protocol is still receipt-free, given that the voting servers

are trusted and do not reveal the voting serial number and corresponding tag.
- *Forced-abstention.* Our scheme is secure against forced-abstention attacks since the voter can authenticate by a zero-knowledge proof and then vote with a tag. Furthermore, the ballot including the tag information does not identify the voter to the adversary.
- *verifiability.* The tallying phase is verifiable and efficient.

?? Our scheme addresses the problem of flooding the bulletin board with an excessive number of valid ballots. Both voters and vote buyers are aware that only the first ballot of a voting credential is valid. In addition, our scheme does not count the total number of ballots cast by a voter in the tally phase. Therefore, there is no courage to vote multiple times.

1.2 Related work

Benaloh and Tuinstra [3] proposed the first scheme to achieve receipt-free voting. Their idea is extended by Sako and Kilian [27] that applies mix-networks which shuffle the order of encryption of yes/no votes and then send the order to the voter through an untappable channel. Receipt-freeness property based on untappable channel with various cryptographic scheme and efficiency were introduced in [24, 17, 26] and in the coercion-resistance scheme [18, 5].

In BeleniosRF [9], there is a trusted randomization server to provide receipt-freeness, and it is not secure against forced-abstention attacks. The voting schemes proposed in [20, 22] allow multiple votes to be submitted and take the sum of all votes as the final results. The method to achieve receipt-freeness in [20] relies on dummy ballots, while in [22] the receipt-freeness with everlasting privacy is achieved by using mix-networks, commitment schemes, and zero-knowledge proofs. These receipt-freeness schemes are not secure against forced-abstention attacks.

Everlasting privacy protects the voter's privacy against a computationally unbounded adversary. Moran and Noar [23] introduced a verifiable receipt-freeness scheme with everlasting privacy using an untappable channel that models a private polling booth. An enhancement of Helios and Prêt à voter with everlasting privacy towards the public are presented in [14, 13]. These works' main techniques to achieve everlasting privacy are perfectly hiding and computationally binding commitment schemes and untappable channels. In some proposals for protocols with everlasting privacy, it is assumed that a private channel exists between the voter and the voting server where the voting server is trusted [14, 7, 13]. Locher and Haenni [22] addressed this limitation as their everlasting privacy does not rely on trusted authorities and computational assumption but not for receipt-freeness. In their scheme, the adversary is passive and not secure against forced-abstention attacks. Furthermore, their protocol requires an extra trust assumption to remain secure against the attacks in which the adversary requires a certain number of the receipt from the voter (i.e., 1009 attack).

1.3 Paper overview

2 Cryptographic preliminaries

In this section, we introduce the cryptographic building blocks that we use. For more detail, see [19].

2.1 Encryption, mix-network and commitment schemes

ElGamal Encryption Scheme. The ElGamal encryption scheme is a triple of algorithms `(KeyGen, Enc, Dec)` that define as follows:

- $\texttt{KeyGen}(\lambda) \rightarrow \mathcal{PP}$: on input a security parameter λ outputs a public parameter $\mathcal{PP} = (\mathcal{G}, q, g)$ $\mathcal{PP}$ denotes cyclic group $\mathcal{G}$ of prime order q generated by g and an encryption key pair (pk, sk).
- $\texttt{Enc}(\mathcal{PP}, m, pk) \rightarrow (c_1, c_2)$: on input $\mathcal{PP}$, a message m, and pk, outputs a ciphertext (c_1, c_2), where $c_1 = g^r$, $c_2 = mpk^r$, and $r \in_U \mathbb{Z}_q$. We denote uniformly at random with notation $\in_U$. In our scheme the message m is of form $m = g^a$ where $a \in \mathbb{Z}_q$.
- $\texttt{Dec}((c_1, c_2), sk) \rightarrow m$: on input a ciphertext (c_1, c_2) and secret decryption key sk, outputs $m = c_2 \cdot c_1^{-r}$.

The ElGamal encryption scheme satisfies the semantic security under the Diffie-Hellman assumption. We denote pk_T the election public encryption key and decryption key sk_T, where $pk_T = g^{sk_T}$.The decryption key can be distributed using (t, n) Shamir secret sharing so that $t > 1$ out of n shares are required to decrypt the ciphertext [6]. A re-encryption of a given ciphertext (c_1, c_2) with a new $r' \in_U \mathbb{Z}_q$ can be simply computed by multiplying to a cipherthext of the encryption of zero i.e g^0 (or $m = 1$). The $reEnc(\mathcal{PP}, (c_1, c_2), pk) = (c_1, c_2) \cdot Enc(\mathcal{PP}, g^0, pk)$. The re-encryption result of the given ciphertext can be described with $Enc(g^m, pk)$ with a randomness $r + r'$. We use *Re-encryption mix servers* in the tally phase. A distributed mixing protocol for a given set of El-Gamal ciphertexts $\{(c_1, c_2)_1, (c_1, c_2)_2, \ldots, (c_1, c_2)_n\}$ and a uniformly random, secret permutation ρ, outputs $\{(c'_0, c'_1)_{\rho(1)}$,
$(c'_0, c'_1)_{\rho(2)}, \ldots, (c'_0, c'_1)_{\rho(n)}\}$. It can be done by applying re-encryption method and it is infeasible for a computationally bounded adversary to match inputs and outputs. Indeed, $(c'_0, c'_1)_{\rho(i)}$ is a sequence of the re-encryption $(c_1, c_2)_i$ for a sequence secret permutations and random re-encryption. To achieve verifiability, each server performing the mix can generate a proof that verify that the output of the mix satisfies with the given input and public key. With re-encryption mix servers, we can achieve privacy if there is only one honest mix server. In this case, no one can find a link between the output ciphectext of the mix and the input one with the advantage better than guessing.

Commitment Scheme. A commitment scheme is a triple of algorithms `(Setup, Commit, VerCommit)` that define as follows:

- $\texttt{Setup}(\lambda) \rightarrow \mathcal{PP}$: on input a security parameter λ, outputs public parameters $\mathcal{PP}$ as a desdription of the message space, commitment space, and commitment key space.
- $\texttt{commit}(\mathcal{PP}, m) \rightarrow (com, r)$: on input $\mathcal{PP}$ and a message m, outputs a commitment com to message m and opening randomness r.
- $\texttt{VerCommit}(com, m, r) \rightarrow 0/1$: on input a acommitment com on message m with randomness r, outputs 1 if accept and 0 otherwise.

Commitment schemes satisfy the correctness, binding and hiding properties [25]. In our scheme we can use two types of the commitment schemes: *Pedersen commitments* [25] and *hash commitments*. The Pedersen commitments are perfectly hiding, computationally binding, and additively homomorphic. The commitment schemes with a collision-intractable hash functions are statistically-hiding and computationally binding. We denote the hash commitment $com = H(k, r)$ for commitment to a message k with randomization r. (note???. We will use hash commitment scheme as the Pedersen commitments version can be find in [22])

2.2 Signature schemes and zero-Knowledge proofs

Digital Signature Scheme. A digital signature scheme is a triple of algorithms `(KeyGen, Sign, verify)` that define as follows:

- $\texttt{KeyGen}(\lambda) \rightarrow (sk_\sigma, pk_\sigma)$: on input a security parameter λ outputs a signing key pair (sk_σ, pk_σ). We let pk_σ denotes the signature verification key and sk_T, the signature secret key.
- $\texttt{Sign}(sk_\sigma, m) \rightarrow \sigma$: on input a message $m \in \{0,1\}^*$, a secret signing key sk_σ, outputs a signature σ on message m.
- $\texttt{Verify}(\sigma, m, pk_\sigma) \rightarrow 0/1$: on input a signature σ on message m and signature verification key pk_σ, outputs outputs 1 if accept and 0 otherwise.

A digital signature scheme satisfies the correctness and existential unforgeability properties [19].

In our scheme we also use *Threshold Signature Scheme*, a threshold signature scheme is a method that replaces the key generation and signature algorithms of a digital signature scheme with an interactive protocol amongst multiple parties. That is, a set of n parties interactively generate a t-out-of-n secret sharing of the key. Then, an interactive protocol is used to generate a signature through the signing algorithm. If $t > 1$ parties agree on a message to sign, then they can generate a signature. However, no subset smaller than t can generate a signature on a message [21, 15]. It should be noted that the verification algorithm remains unchanged.

Zero-Knowledge Proofs. Let $\{R_\lambda\}_{\lambda \in \mathbb{N}}$, λ is security parameter, be a family of polynomial-time decidable relations R on pairs (x, w) where x is the statement, and w is the witness. We denote $R(x, w) = 1$ to show that (x, w) satisfies on R, $R(x, w) = 0$ otherwise. A Non-Interactive Zero-Knowledge (NIZK) for $\{R_\lambda\}_{\lambda \in \mathbb{N}}$ is a triple of algorithms `(KeyGen, Prove, Verify)` that define as follows:

- $\texttt{KeyGen}(R, \lambda) \rightarrow (ek, vk)$: on input a security parameter λ and a relation $R \in \{R_\lambda\}_{\lambda \in \mathbb{N}}$, outputs a common reference string consisting of an evaluation key ek and a verification key vk.
- $\texttt{Prove}(ek, x, w) \rightarrow \pi$: on input ek, an evaluation key for a relation R, a statement x, and a witness w such that $R(x, w) = 1$, outputs a proof π.
- $\texttt{Verify}(vk, x, \pi) \rightarrow 0/1$: on input vk a verification key, a statement x, and a proof π outputs 1 if accept the proof and 0 otherwise.

NIZKs satisfy the completeness, knowledge soundness and zero-knowledge properties. The zero-knowledge Succinct Non-interactive Arguments (zkSNARKs) [2] is a type of NIZKs with additional property, succinctness, i.e. the verification time is $\texttt{poly}(\lambda + |x| + \log |w|)$ and the proof size is $\texttt{poly}(\lambda + \log |w|)$.

Zero-knowledge proofs are the main tool in our protocol to achieve the voter's privacy. In our scheme, the Zk-SNARK enables a voter to prove that he knows the pre-image k and the commitment randomness r such that the voting serial number is equal to $H(k)$ and $com := H(k, r)$). However, nobody knows that the voting serial number is assigned to which commitment in the voters' commitment list. In addition to prove the eligibility of voter, the Zk-SNARK proof protects the unlikability between the voting serial number and the voter's identity.

We also use the NIZKs in the following cases: the voter provides proof of knowledge of the secret randomness and the candidate code involved in the encryption of the vote. The voter's signature on the ballot also ensures the knowledge of the related secret signing key and the integrity of the ballot. In the tally phase, the NIZKs prove the validity of mixing and decryption [16, 10, 17, 28, 8].

It is worth to note that the zk-SNARK requires a one-time trusted setup of public parameters i.e., (ek, vk). This trust assumption violation might affect the soundness of the proofs though privacy continues to hold even if the setup trapdoors can be computed by quantum computers. In [1], the authors proposed the first transparent zero-knowledge system that the set up does not rely on any trusted party, and has no trapdoors that could be exploited by powerful parties to prove false witness.

3 Privacy preserving receipt-free e-voting

3.1 Adversary model and trust assumptions

We first Rosario: why don't we just consider only the receipt-free one? consider a polynomial bounded adversary Rosario: shouldn't it be a quantum one? whose goal is to break the correctness or privacy of the votes during an election by submitting votes on behalf of someone else or by linking votes to voters. The cryptographic primitives such as encryption, commitments, collision-resistant hash functions, and zero-knowledge proofs rely on hard mathematical assumptions. Therefore, a polynomial bounded adversary has a negligible advantage in impersonating, altering, or canceling the votes of honest voters.

Another type of adversary is a vote buyer. A vote buyer aims to pay rewards to dishonest voters who can convince the adversary with a receipt. The receipt contains information to show how the voters voted. In general, receipt-freeness does not prevent an adversary from buying the voter's private key and voting on behalf of the voter. A vote buyer adversary in our protocol can even cast a vote with the knowledge of the vote serial number. Furthermore, an adversary can coerce voters to not participate in voting (i.e., forced-abstention attacks).

Our protocol is receipt-free under the following assumptions on adversarial capabilities: we assume the adversary is computationally limited and not monitoring the interaction between the voter and the voting servers. The voter can cast a ballot any time before providing the receipt. The (majority) voting server(s) and tallier(s) are honest. The bulletin board, voting device, and registrar are all trusted. A trusted registrar publishes a list of registered voters on the bulletin board without removing any registered voters.

3.2 Protocol description

We provide a high level description of our protocol Rosario: I think we should go for a detailed explanation and explain its receipt-free property.

Protocol participants. Our protocol consists of the following participants: Election Authorities $\mathcal{EA}$, Registrar $\mathcal{R}$, Voters $ID = \{id_1, id_2, \ldots, id_m\}$, Voting Servers $\{\mathcal{VS}_1, \mathcal{VS}_2, \ldots, \mathcal{VS}_n\}$, (or $\mathcal{VS}$), Talliers $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n\}$ and an append-only bulletin board $\mathcal{BB}$.

- *Setup.* In the *setup phase*, the Election Authorities $\mathcal{EA}$ on input security parameter 1^λ, threshold tuple (t, n), candidate list $\mathcal{C}$ jointly generate the encryption parameters $(\mathcal{G}, q, g)$ election encryption key pair (pk_T, sk_T), a unique identifier g^{a_i} for each candidate c_i, a signature key pair for voting server(s) $(pk_{\mathcal{VS}}, sk_{\mathcal{VS}})$. The $\mathcal{EA}$ select a collision-resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^{O(\lambda)}$ with the corresponding prover and verifier key pair (ek_H, vk_H). The $\mathcal{EA}$ publish the hash function H, (ek_H, vk_H), pk_T, $pk_{\mathcal{VS}}$ and public parameters, and candidate list $\mathcal{C}$ with their code on the public bulletin board $\mathcal{BB}$.
- *Registration.* A voter during the registration generates a commitment to her secrete key and submits to a registrar $\mathcal{R}$ Rosario: can we merge EA and R? It would make more senses via an authentic channel. A voter commitment is $com_{id} := H(r_{id}, k_{id})$ where r_{id} is an opening value to the committed value k_{id}. The registrar publishes a list of the eligible voters *VList* on the bulletin board $\mathcal{BB}$. *VList* contains the identity of the voter and corresponding commitment $VList = \{(com_{id_1}, id_1), \ldots, (com_{id_m}, id_m)\}$. A registered voter can verify the published commitment next to his identity on the $\mathcal{BB}$.
- *Voting.* The *voting phase* is as follows:
  - *Voter Authentication*: In this phase, the distributed *voting servers* issue a voting tag to an eligible voter to cast her vote.
    A voter sends the voting serial number $H(k_{id})$ along with a zero-knowledge proof π_H to the voting servers through an anonymous channel. The voting servers issue a tag based on their voting serial number list *SList*.

The $\mathcal{VS}$ jointly issue a valid tag if there is no copy of $H(k_{id})$ in their $SList$ and invalid tag otherwise.
A valid tag Tg is a tuple of encryption of 1, i.e., $E_1 := Enc(g^1, r_1, pk_T)$ and a unique signing key pair $(sk_v, pk_v)_{\sigma_v}$ that is signed by the threshold signature of the voting servers $\sigma_{\mathcal{VS}}$, namely $Tg = ((E_1, pk_v)^{\sigma_{\mathcal{VS}}}, sk_v)$. An invalid tag is of form $Tg = ((E_0, pk'_v)^{\sigma_{\mathcal{VS}}}, sk'_v)$ where $E_0 := Enc(g^0, r_0, pk_T)$.
The voting servers hide the number of the votes from the user whether the voter has already voted or not. The voter privacy is achieved by the zero-knowledge property of the proof and the serial number $H(k_{id})$ which does not leak any information regarding the public commitment of the voter. Precisely, finding which commitment in the $VList$ corresponds to a particular voting (serial) number is equivalent to finding a preimage of the hash value $H(k_{id})$ and commitment randomness r_{id}, which is assumed to be infeasible. Thus, Tg, $H(k_{id})$, and π_H does not leak the identity of the voter.

- *Casting.* To cast a vote, the voter who has the Tg as a private input, encrypts her candidate with a zero-knowledge proof of correct encryption (c_1, c_2, π_c) where $(c_1, c_2) := Enc(g^{a_i}, pk_T)$, randomness $r \in_U \mathbb{Z}_q$ and $g^{a_i} \in \mathcal{C}$.
  The voter generates a ballot $\beta = (c_0, c_1, \pi_c, (E_{0/1}, pk_v)^{\sigma_{\mathcal{VS}}}, \sigma_{pk_v})$ where σ_{pk_v} is a signature of the voter on the information of the ballot using his secret key sk_v. The voter submits the ballot β to the $\mathcal{BB}$. The $\mathcal{BB}$ checks the validity of the signature $\sigma_{\mathcal{VS}}$ on $(E_{0/1}, pk_v)$ then verifies that neither the ciphertext (c_1, c_2) nor $(E_{0/1}, pk_v)$ appear in any ballot on the $\mathcal{BB}$. The $\mathcal{BB}$ publishes the ballot β on he $\mathcal{BB}$ if β be valid with all the checks, otherwise returns $\perp$. The voter can verify the casting ballot on the $\mathcal{BB}$.

– *Tally.* In the *tally* phase, all ballots with valid proofs are selected, signatures and the public signing keys are removed by talliers. Then, the ballots are shuffled and mixed. The $E_{0/1}$s of the mixing ballots are decrypted by talliers then the valid ballots that correspond with decryption of the E_1's e.g., g's are homomorphically multiplied. In the final step, the talliers decrypts the resulted ciphetext and publish the result with the correctness proof of the decryption. The process of tallying can be done inverse, namely the tallier first decrypt the balllots, sort the candidates with the corresponding $E_{0/1}$, and multiply all $E_{0/1}$ next to each candidate. The number of votes for each candidate is the decryption of the multiplied ciphertext.

4 Security properties

We informally discuss the security properties of our proposed scheme in the following sections.

4.1 Ballot privacy

Ballot privacy is the property of not being able to learn more from election public information than from the results alone. A computationally bounded adversary

against ballot privacy would be a voter(s) itself or an outside observer. This property hold under the assumptions that the voting device is secure, bulletin board and majority of talliers are honest.

4.2 Verifiablity

Our proposed protocol is verifiable in the following sense: each voter can check that their registration by verify the commitment in the public bulletin board. A voter can verify that her vote is included in the tally. The correctness of the final tally is verifiable. Anyone can verify that all votes cast are included in the tally phase, only valid votes are counted, and that no votes are altered during tallying.

4.3 Double voting detection

Our scheme detects double voting based on the serial number that voters submit to the voting servers. Prior to issuing the tag, the distributed voting servers check the $SList$ to verify that there is no duplicate serial number. In other words, the voting servers jointly generate a valid tag (encryption of 1) for the first serial number that is appended to their $SList$ and an invalid tag (encryption of 0) in the event that there is a douplicate of the same serial number. These tags in the tallying phase are decrypted to count the votes with valid votes.

4.4 Receipt-freeness

We discuss various kind of receipts that a vote buyer may accept in our protocol.

- **A voter absentee and voting serial number.** An attacker can coerce the voter to abstain from voting. In this case, a voter can vote, as the ballot has no information about the voter's identity and is unlikable to the voting serial number. Later, a voter who casts her vote can give the attacker the voting serial number to cast a vote. An attacker can not distinguish whether a voter voted or not from the $\mathcal{BB}$ and the voter secret keys. We assume that the voting servers are honest and hence they do not reveal any information about the voting serial numbers. Our schemes can be described that a voter can give her secret keys to the attacker any time but not prior to the voting phase.
- **Encryption randomness and tag.** A voter may vote multiple times with valid proofs and various tags during the voting phase, and all of them will publish and include in the tally phase. Therefore, the randomness of the encryption of a vote and tag can be given to the vote buyer as a receipt. This receipt does not reveal how the voter voted as the tag hides the information about how many times the voter has voted. However, the voter can not cast multiple ballots with the same tag or modified tag. Since there is a signature of the $\mathcal{VS}$ on $E_{0/1}$. If a ballot is submitted to $\mathcal{BB}$ using the same tag of an

existing ballot on $\mathcal{BB}$ then the bulletin board will refuse to accept the second one.

We assume that because of the security of the signature and encryption schemes it is infeasible for a polynomially bounded adversary to forge a signature scheme or provide a proof of correct encryption for the ciphertext of other voters.

– **A specific and large number of ballots.** A vote buyer may want a receipt (encryption randomness or tag) for a large number ballots (1009 attack). In this case, the attacker want to verify the number of zero decryption in the tally phase. For instance, if there are 1008 zeros in the first tally then one can conclude that one vote is accepted as a valid vote. Our scheme provides counter-measure to this type of receipt using the mixing and shuffling features. By mixing the ballots, the attacker can not verify the ballots in the tally phase by the encryption randomness. In the case of the number of tags, a voter can vote more that 1009 or the extra zero might be generated by other votes.

5 Conclusion

This paper proposes a verifiable receipt-free e-voting protocol with distributed voting servers that maintains the privacy of voters against adversaries with a variety of capabilities. We provide receipt-freeness property for adversaries involved in the election process, and voter privacy for computationally powerful adversaries in the future.

note to myself: For a distributed voting authorities setting, there is no need for proof of the tag. Does the voter have to vote via an untappable channel like a voting booth? If the adversary does not control the voter in the voting phase, the voter, by providing the receipt, can not convince the adversary. In case there is a distributed voting authority, and the voter has first voted, the voter can give the adversary her secret information to vote on behalf of the voter or vote in the presence of the adversary.

We provide the privacy of the voter by breaking the link between the voter registration public key and the voter voting credential. The voting serial number is unique and can be connected with the voter commitment. Using a zero-knowledge proof, the voter proves the connection, but nobody knows which voting serial number is assigned to which commitment in the commitment list.

Protocol properties:

– The privacy property of e-voting protocols that voters is identified with their digital signature on the ballot of the voter are vulnerable against the quantum computer computational power.
– Anonymous voting using the zero-knowledge SNARK or its post-quantum version STARK. BeleniousRF is not secure against vote abstain attack, even if the voter selects his secret key during the voting, the signing public key

reveals the voter's identity. In our scheme the voter remains anonymous during the voting phase.
- Receipt-freeness using the designated proof of knowledge(non-transferable proof) or distributed setting voting server.
- the vote buyers are not passive during the voting period, i.e., they can cast votes on behalf of the voters. Our protocol is secure against active adversaries running impersonation or forced-abstention attacks.
- The voter can convince the buyer by providing the randomness of the encryption of the vote and simulated proof of the tag.
- An adversary is assumed not to monitor the interaction between the voter and the voting server in the definitions of the receipt-freeness voting scheme. However, the voter can record this interaction and present this information (or any transformation thereof) to the adversary (from BeleniosRF).
- the proposed scheme can be seen as a receipt-freeness under the assumption that the voting servers, which are in charge of sending the vote credentials (tags), are trusted to generate tags related to the vote serial number based on double vote policy and not reveal any information to any party about the tag (except a designated proof used for that procedure).

References

1. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive (2018)
2. Ben Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from bitcoin. In: 2014 IEEE Symposium on Security and Privacy. pp. 459–474 (2014). https://doi.org/10.1109/SP.2014.36
3. Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections. In: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing. pp. 544–553 (1994)
4. Bernhard, D., Kulyk, O., Volkamer, M.: Security proofs for participation privacy, receipt-freeness and ballot privacy for the Helios voting scheme. In: Proceedings of the 12th International Conference on Availability, Reliability and Security. pp. 1–10 (2017)
5. Bohli, J.M., Müller-Quade, J., Röhrich, S.: Bingo voting: Secure and coercion-free voting using a trusted random number generator. In: International Conference on E-Voting and Identity. pp. 111–124. Springer (2007)
6. Brandt, F.: Efficient cryptographic protocol design based on distributed el gamal encryption. In: International Conference on Information Security and Cryptology. pp. 32–47. Springer (2005)
7. Buchmann, J., Demirel, D., Graaf, J.v.d.: Towards a publicly-verifiable mix-net providing everlasting privacy. In: International Conference on Financial Cryptography and Data Security. pp. 197–204. Springer (2013)
8. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: Annual International Cryptology Conference. pp. 410–424. Springer (1997)
9. Chaidos, P., Cortier, V., Fuchsbauer, G., Galindo, D.: Beleniosrf: A non-interactive receipt-free electronic voting scheme. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 1614–1625 (2016)

10. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Annual international cryptology conference. pp. 89–105. Springer (1992)
11. Chow, S.S., Liu, J.K., Wong, D.S.: Robust receipt-free election system with ballot secrecy and verifiability. In: NDSS. vol. 8, pp. 81–94 (2008)
12. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a secure voting system. In: 2008 IEEE Symposium on Security and Privacy (sp 2008). pp. 354–368. IEEE (2008)
13. Demirel, D., Henning, M., Graaf, J.v.d., Ryan, P.Y., Buchmann, J.: Prêt à voter providing everlasting privacy. In: International Conference on E-Voting and Identity. pp. 156–175. Springer (2013)
14. Demirel, D., Van De Graaf, J., dos Santos Araújo, R.S.: Improving helios with everlasting privacy towards the public. Evt/wote **12** (2012)
15. Doerner, J., Kondi, Y., Lee, E., Shelat, A.: Secure two-party threshold ecdsa from ecdsa assumptions. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 980–997. IEEE (2018)
16. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Conference on the theory and application of cryptographic techniques. pp. 186–194. Springer (1986)
17. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 539–556. Springer (2000)
18. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. Association for Computing Machinery, New York, NY, USA (2005)
19. Katz, J., Lindell, Y.: Private key encryption and pseudorandomness. Introduction to Modern Cryptography, Chapman & Hall/CRC Cryptography and Network Security pp. 47–109 (2007)
20. Kulyk, O., Teague, V., Volkamer, M.: Extending helios towards private eligibility verifiability. In: International Conference on E-Voting and Identity. pp. 57–73. Springer (2015)
21. Lindell, Y., Nof, A.: Fast secure multiparty ecdsa with practical distributed key generation and applications to cryptocurrency custody. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 1837–1854 (2018)
22. Locher, P., Haenni, R.: Receipt-free remote electronic elections with everlasting privacy. Annals of Telecommunications **71**(7), 323–336 (2016)
23. Moran, T., Naor, M.: Receipt-free universally-verifiable voting with everlasting privacy. In: Annual International Cryptology Conference. pp. 373–392. Springer (2006)
24. Okamoto, T.: Receipt-free electronic voting schemes for large scale elections. In: International Workshop on Security Protocols. pp. 25–35. Springer (1997)
25. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Annual international cryptology conference. pp. 129–140. Springer (1991)
26. Ryan, P.Y., Rønne, P.B., Iovino, V.: Selene: Voting with transparent verifiability and coercion-mitigation. In: International Conference on Financial Cryptography and Data Security. pp. 176–192. Springer (2016)
27. Sako, K., Kilian, J.: Receipt-free mix-type voting scheme. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 393–403. Springer (1995)
28. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of cryptology **4**(3), 161–174 (1991)