VoteXX: Coercion Resistance for the Real World (preliminary extended abstract)

David Chaum[1], Richard T. Carback[1], Jeremy Clark[2], Chao Liu[3], Mahdi Nejadgholi[2], Bart Preneel[4], Alan T. Sherman[3], Mario Yaksetig[1], Filip Zagórski[5], and Bingsheng Zhang[6]

[1] xx.network, USA, david@chaum.com
[2] Concordia pulpspy@gmail.com
[3] Cyber Defense Lab, Univ. of Maryland, Balt. County (UMBC), Baltimore, MD 21250, USA, sherman@umbc.edu
[4] COSIC, KU Leuven and imec, Belgium, bart.preneel@esat.kuleuven.be
[5] Poland filip.zagorski@gmail.com
[6] Zhejiang University, Hangzhou, China, bingsheng@zju.edu.cn

Abstract. We solve the most challenging obstacle to casting votes online: "improper influence," typically defined as vote buying or voter coercion. This problem has been the greatest barrier to mainstream adoption of online voting systems, whether the voter receives the ballot online or by mail. Our conceptual breakthrough allows each voter, or their trusted associates (which we call "hedgehogs"), to *"nullify"* (cancel) their vote in a way that is anonymous, unstoppable, and irrevocable. The separate nullification stage requires fewer assumptions than do other approaches, thereby strengthening security and increasing practicality, while simplifying and speeding up registration and vote casting. Though our nullification technology can be readily applied to almost all types of elections for which the tally is publicly auditable online, its benefits are greatest for votes cast beyond polling places, where the threat of improper influence is most acute. It addresses current trends: elections are moving online despite the growing ease by which voters can be paid and coerced electronically. Nullification will help make it prudent to vote online, which is the type of voting advocates have long believed needed for democracy to reach its potential.

We introduce the new approach, give detailed cryptographic protocols realizing it, show how it can be applied to several voting settings, and describe our implementation of it. The protocols compose a full voting system, which we call *VoteXX*, including registration, voting, nullification, and tally—using the cMix mix network for untraceable communication including vote casting. We demonstrate how the technique can be applied to known systems, including Remotegrity, where ballots can be mailed to voters and voters use codes on the ballot to cast their votes online. We also present progress toward a proof of security in the UC framework. As part of a broader survey of 537 papers on voting technologies, we performed LDA analysis of the 227 papers that discuss coercion-resistance. Because an adversary could coerce a voter not to register or vote, nullification achieves the optimal possible protection against a strong coercer who learns all of the voter's secrets. We offer a flexible solution to improper influence using the realistic assumption of an untappable channel between the voter and their hedgehogs.

In VoteXX, each voter has two Diffie-Hellman public-private key pairs, one for "YES" votes, and one for "NO" votes. Without revealing their private keys, each voter registers their public keys with the election authority. Each voter may share their keys with one or more hedgehogs. During nullification, the voter, or one or more of their hedgehogs, can interact with the mix network to nullify a vote by proving knowledge of one of the voter's private keys via a zero-knowledge proof without revealing the private key. We describe a fully decentralizable implementation of VoteXX, including its public bulletin board, which could be implemented on a blockchain.

Keywords: cMix · coercion resistance · decentralized election authority · election security · Elixxir Network · hedgehog · high-integrity voting system · Internet voting · mixnet · mix network · nullification · remote voting · VoteXX · xx.network.

1 Introduction

For decades, researchers have attempted without success to design a voting system suitable for use through the Internet. Three daunting challenges make Internet voting difficult: (1) The lack of a secure physical voting precinct facilitates improper influence, including vote selling and coercion. (2) Malware on the voter's device (e.g., phone) might undetectably modify votes and spy on voters. (3) Determined adversaries might try to launch an online attack, including causing outages. Of these challenges, the most elusive has been mitigating improper influence.

We present and analyze a practical solution to this elusive problem. A key feature of our approach is to separate the mechanism for mitigating improper influence from the mechanisms for marking and collecting ballots. Consequently, our approach can be applied to a wide variety of voting systems, including precinct voting with paper ballots, voting by mail, and Internet voting. This separation also facilitates adopting best practices for the rest of the system, including, for example, *End-to-End (E2E)* voter verification [23].

We present the *VoteXX* voting system, which embodies a new and effective mitigation to improper influence based on the novel concept of *"vote nullification"* (also called *"vote flipping"*). The voter, or one of their designated trusted associates (called *"hedgehogs"*), can cancel the vote during a "*nullification period*" that takes place after vote casting and before vote tallying. We prove that nullification achieves the theoretically best possible response to improper influence. We explain how this mitigation can be applied in a variety of voting contexts, including precinct voting with paper ballots, voting by mail, and Internet voting. Our implementation of VoteXX running on the xx.network [30] features a fully decentralizable *Election Authority (EA)*.

The *VoteXX Project* imagines a future in which citizens can vote on their smart phones with security and privacy. Such remote Internet voting offers numerous advantages: convenience, low cost, more accurate ballot marking, fast reporting of results, and improved usability and accessibility, including support for multiple languages and long ballots.

Although some countries have already forged ahead despite security and privacy vulnerabilities, some leading computer scientists—including MIT's Ronald Rivest [15]—admonished in 2010 that, given current limitations of technology, Internet voting is akin to "drunk driving."

Recent examples showing the limitations of Internet voting bear out this opinion. For example, Lewis et al. [17–19] show that in Switzerland's SwissPost-Scytl sVote voting system, a corrupt system could change valid votes into invalid votes. Teague [28] explains that the iVote system used in New South Wales, Australia, has the same weakness. Specter et al. [26] point out serious vulnerabilities in the Voatz blockchain system used in West Virginia, U.S., enabling adversaries to modify and expose votes cast from mobile phones. Our approach, however, overcomes these limitations.

Widely used mail-in ballots are not susceptible to malware attacks at the voter's end, but they too are vulnerable to improper influence. For example, a dishonest or coerced voter might take a video of a voter filling out, sealing in an envelope, and mailing their ballot. Similarly, a voter could also covertly take a video of themself voting in a precinct—curiously, in some precincts such videos are even permitted. It should not be possible for a voter to prove how they voted.

Previous attempts to mitigate improper influence complicate registration, vote-casting, or credential management (e.g., assuming an untappable channel between the voter and the registration authority to issue fake credentials [14]), make assumptions that are difficult to realize in practice (e.g., assuming the adversary does not interfere with the last revoting step [29]), or depend on decoy ballots [5]. By contrast, VoteXX offers a flexible solution using the realistic assumption of an untappable channel between the voter and their hedgehogs. For simplicity, we shall use the term "coercion" to include vote selling (cf. Section 2).

Mitigating improper influence is the hardest challenge for Internet voting. In VoteXX, each voter has an option to cancel their vote using a "*flip code*," which they establish during registration. In a two-candidate race, flipping means computing the modulo-two sum of the original vote and any flipped votes for that ballot question. The idea can be generalized to a modulo-k sum for a k-candidate race.

The flip code is the private key of a public-private key pair. During registration, the voter presents the public key but never reveals the private key to the system. To flip a vote, the voter engages in a zero-knowledge proof with the system to prove that they know the private key, without revealing the private key.

The coercer can demand that the voter reveal all of the secret keys the voter possesses, including the flip code. Revealing the flip code to the coercer enables the coercer to flip the voter's vote, but revealing the flip code does not prevent the voter from flipping their vote. Effective coercion cannot take place after the voter casts their ballot.

Our approach also introduces the new concept of hedgehogs, trusted associates of the voter. Optionally, each voter can enlist one or more hedgehogs and reveal the flip code to them. To flip their vote, it would be sufficient for the voter to signal at least one hedgehog. The signal could be subtle and covert—such as moving a specified potted plant on a balcony or posting a picture of a specific object on Twitter. Hedgehogs are useful when a coercer closely monitors the voter. We assume that, at some time after registration and before the nullification period, the voter is able to communicate their flip code to their hedgehog(s) privately. We also assume that, at some time after any coercion, and before the nullification period, the voter is able to signal the hedgehog(s) privately.

Nullification is different from multiple voting. In multiple voting, only the voter can vote, and the coercer might be able to coerce the voter at the end of the voting period. By contrast, in nullification, there might be one or more hedgehogs, and the coercer does not know who they are or how many there are.

Nullification achieves the theoretical limit of what is possible in mitigating coercion: a coercer could simply threaten a voter not to vote or even register to vote. With nullification, improper influence is not possible because the coercer cannot be certain that the voter followed the coercer's demands.

To thwart malware the VoteXX system uses E2E verification: using public audit data, voters verify that their ballots were cast, collected, and counted properly, without revealing how they voted. In this way, the system is "software-independent" [24]: without assuming trust in the correct operation of any hardware or software, voters can detect any error in the tally.

Our work contributes:

1. The idea of vote nullification.
2. The design of the VoteXX fully decentralized remote voting system.
3. An implementation of VoteXX on the xx.network, including two different implementations of the vote-nullification mechanism, one using a mixnet, the other using homomorphic encryption.
4. Illustrative approaches for how to transform voting systems into coercion-resistant ones, for four classes of voting systems: (a) Apollo-style [11] fully-remote electronic voting with two channels (numbers come in one channel, votes go out the other), (b) Remotegrity-style [31] voting (mail out scratch-off cards, return votes by Internet), (c) Mail-in voting, and (d) precinct voting.
5. A more modular design of voting systems that separates marking and collecting ballots from mitigating coercion and malware, enabling our new security mechanisms to be easily applied to a wide variety of voting systems.

Our design concepts—including vote nullification and a fully decentralized EA—reduce pressure off the underlying voting system, making it easier to deal with malware and coercion.

2 Background

Coercion resistance guarantees that each voter may vote freely. Informally, a voting system is *coercion resistant* if and only if no voter can prove to any coercer that the voter cast a counted ballot according to the coercer's instructions. For example, voter Alice cannot prove to a coercer that Alice cast a ballot that was counted for Trump. Coercion resistance guarantees that each voter may vote freely.

Coercion resistance is an extremely important and strong property that is difficult to achieve, especially for remote voting, where there is no physically-secure voting place and the machines used to interact with the voting system are untrusted.

Some people distinguish between two types of improper influence: vote buying and coercion. When a voter sells her vote, she acts voluntarily. By contrast, a coerced voter acts involuntarily.

Coercion resistance is at least as strong as *receipt freeness* and possibly stronger. Receipt freeness [3, 22] usually means that the voter cannot prove how she voted, simply by following the voting protocol. For example, an adversary might contact a voter only after the voter finishes voting. Coercion resistance [13, 21, 16] is possibly stronger in that the coercer may in advance instruct the voter to carry out (or not carry out) certain additional actions that are outside of the voting protocol.

Smyth [25] surveys four definitions of coercion resistance and finds that "coercion resistance has not been adequately formalized." Three of the definitions are too weak, and the general definition by Kösters [16] is complex and too strong. Similarly, there remains some debate on the definition of receipt freeness [9].

Coercion-resistance is typically achieved by one of two approach: The first approach is to allow voters to *revote* (vote again), nullifying any previously cast ballots [20, 27, 29]. This approach assumes the adversary does not coerce the voter at the end of the election period, or cannot otherwise lock the voter out from re-voting (*e.g.,* retain their ID smartcard or change their voting credential). The second approach is to allow voters to create *fake credentials* they can use when voting under coercion [1, 2, 7, 8, 10, 13]. Fake ballots are verifiably removed from the tally without revealing which ballots were fake, and much academic effort has been devoted to reducing the computational cost of this filtering process to a cost linear in the number of ballots.

An important aspect of strong incoercibility is the ability for a voter to use an *untappable channel*. It is generally believed that without an untappable channel, the coercer and voter are indistinguishable and therefore incoercibility is impossible to achieve [12]. Re-voting assumes the channel is present after a coercive action, while the use of fake credentials assumes registration is conducted over an untappable channel.

As part ofour AI-powered literature review of improper influence, we performed a keywordsearch of the voting literature from Google Scholar, IACR, IEEE, and CrossRef,yielding 537 articles on voting. Manual search narrowed the set to 237 articleson "improper influence." Using 30 hours of computing, Latent Dirichelt Allocation (LDA) of the 237 articles yielded four topics of keywords. Running a 4-categorymodel against the original inputs ranked articles for our topic of "improperinfluence."

3 Assumptions and Adversarial Model

The adversary could be anyone—including a voter or an EA trustee, located close or far away from their target. The adversary might be covert or overt. The adversary's goal might include any or all of the following: tamper with the tally, influence a voter's ballot choice through coercion, learn how a voter voted, disrupt or discredit an election.

4 Requirements

We seek a multiparty-secure voting system that can be used for remote voting where there is no physically secure voting place. The system must handle the challenges of possible coercion, denial of service, and untrustworthy computers through which the voters interact with the voting system. We describe the voting system requirements by listing crucial security requirements and additional desirable requirements. We also list axioms about the election process, which are necessary to reason precisely about the requirements and their relationships.

There are three crucial security requirements: (1) publicly verifiable correct tally, (2) incoercible, and (3) unstoppable. These requirements imply all of the other security requirements found in the literature.

5 Architecture

We describe VoteXX in terms of the following entities and elements (see Fig. 1). There are n *voters* $v_1, v_2, \ldots, v_n$ who interact with a publicly-readable *bulletin board (BB)*, which is a distributed ledger such as a blockchain. The writing interactions can take place via an *Anonymous Communication System (ACS)*, such as cMix [6], while the read operations can take place via a direct interaction between the voters and the BB. Each voter may have one or more trusted *hedgehog(s)*. The hedgehogs interact with the BB, either directly or via the mix network. The EA consists of three entities: a *Registration Authority*, a *Voting Authority*, and a *Tallying Authority*. The EA can read and write to a BB. The system comprises a set of auditors who can read from the BB and verify that the operations performed by both the EA and via the mix network are correct.

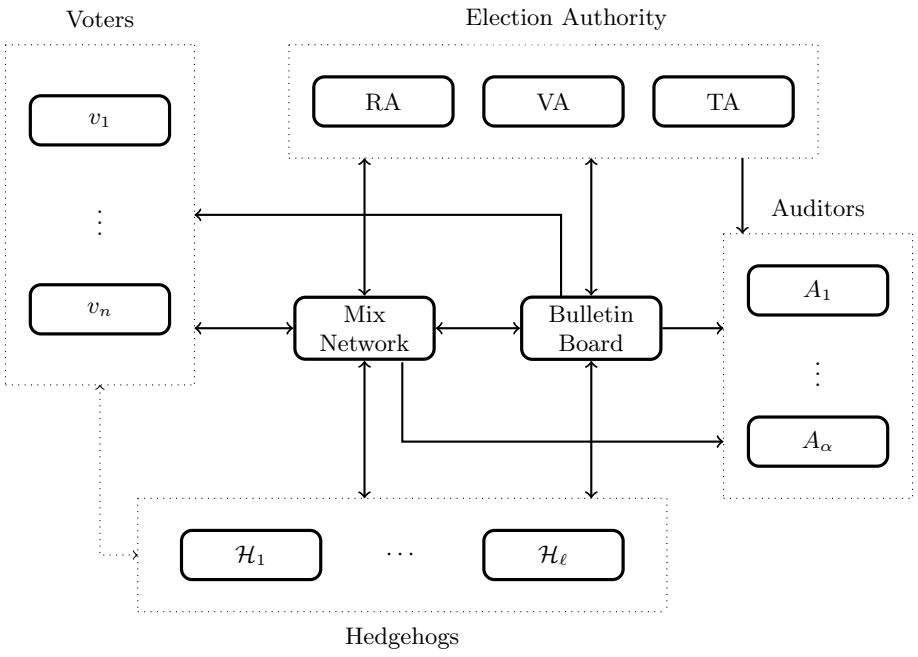

Fig. 1: VoteXX architecture diagram comprising the voters, the hedgehogs, a mix network, a *bulletin board (BB)*, an *election authority (EA)*, and a set of auditors. The arrows represent the information flow (i.e., read/write) between the system entities. Any entity is able to read and write to the BB. Voters, however, use the mix network when writing to the BB to prevent leaking their identity to the system.

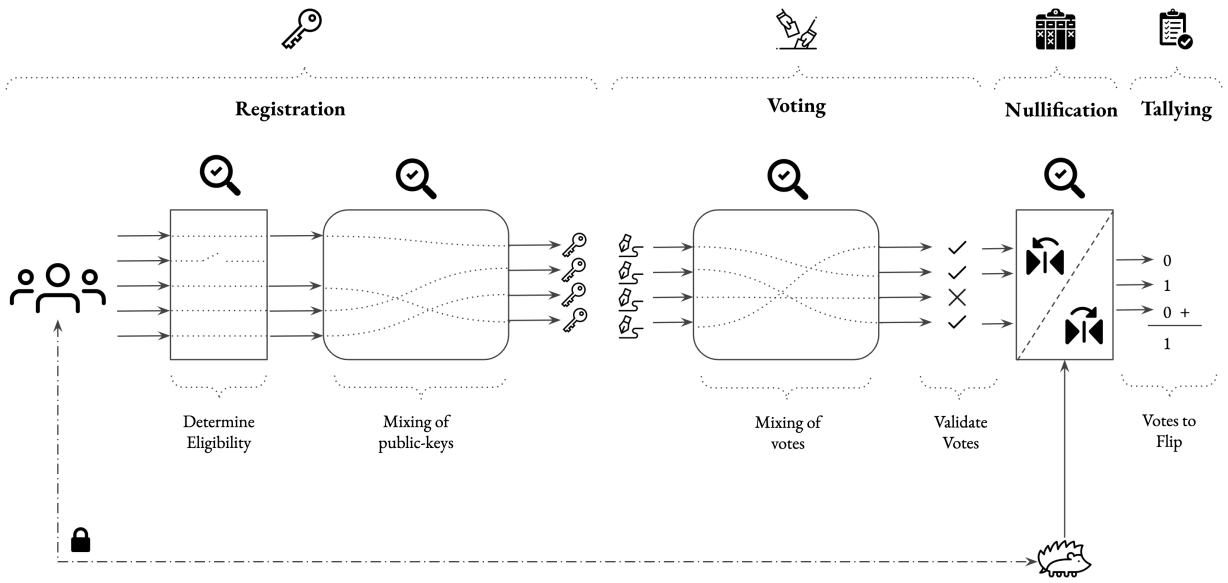

Fig. 2: Overview of the VoteXX protocol, which comprises four stages. First, a registration stage determines the eligibility of each voter. Using a mixnet to hide their identity, each eligible voter registers the public keys for two public-private key pairs—one for "YES" votes, and one for "NO" votes. Second, during the voting stage, each voter casts their (encrypted) ballots by submitting via the mixnet a signature using one of the registered keys corresponding to "YES" or "NO". Third, during the nullification stage, voters or hedgehogs can submit to any node in the mixnet a zero-knowledge proof of knowledge of a voting key to void the corresponding cast ballot. Fourth, the EA computes and publishes a final tally.

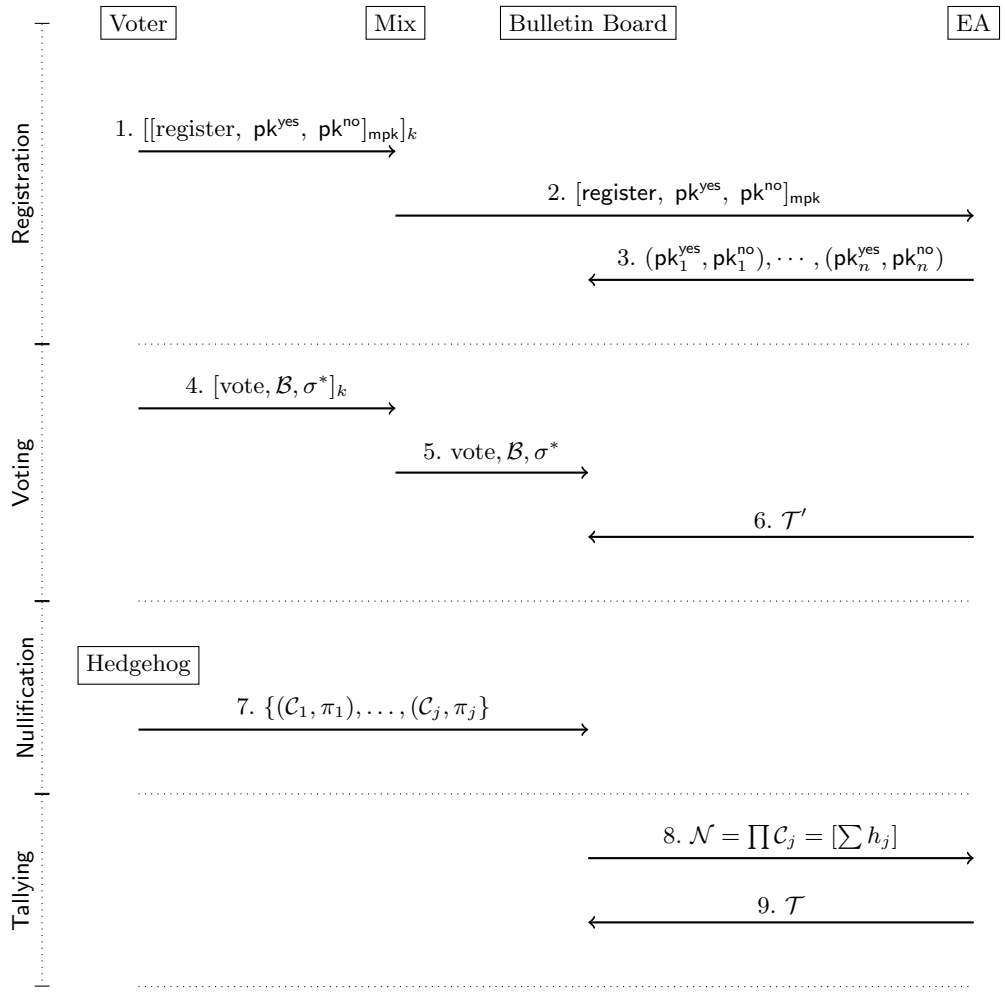

Fig. 3: VoteXX Protocol.

6 VoteXX Protocol

Registration

Prior to registration, each eligible voter performs the following steps:

1. The voter v_i generates two public-private key pairs $(\mathsf{pk}^{\mathsf{no}}, \mathsf{sk}^{\mathsf{no}})$ and $(\mathsf{pk}^{\mathsf{yes}}, \mathsf{sk}^{\mathsf{yes}})$:

   $\mathsf{sk}^{\mathsf{no}} \xleftarrow{\$} G$, $\mathsf{pk}^{\mathsf{no}} = g^{(\mathsf{sk}^{\mathsf{no}})} \mod q$

   $\mathsf{sk}^{\mathsf{yes}} \xleftarrow{\$} G$, $\mathsf{pk}^{\mathsf{yes}} = g^{(\mathsf{sk}^{\mathsf{yes}})} \mod q$

Each voter registers with the EA via a mixnet:

1. Each voter v_i sends $[[\text{register}, \mathsf{pk}^{\mathsf{yes}}, \mathsf{pk}^{\mathsf{no}}]_{EA}]_{k_i}$ to the EA, where k_i is the symmetric key shared with the mix network.
2. The mixnet decrypts the ciphertexts and, after mixing, relays the encrypted message containing the registration of the two public keys $[\text{register}, \mathsf{pk}^{\mathsf{yes}}, \mathsf{pk}^{\mathsf{no}}]_{\mathsf{mpk}}$ to the EA.

After all voters have registered and before voting opens, the EA performs the following steps:

1. The EA posts to the BB a final roster of the two public keys $(\mathsf{pk}^{\mathsf{yes}}, \mathsf{pk}^{\mathsf{no}})$ for each eligible voter.

Protocol 1. Registration.

Voting

To cast a ballot, a voter performs the following steps:

1. For each ballot item $\mathcal{B}$, each voter v_i constructs their vote as $v_i = \sigma_{\mathsf{i}}^{\mathsf{yes}}$ or $v_i = \sigma_{\mathsf{i}}^{\mathsf{no}}$.
2. Each voter v_i sends their encrypted vote $[\text{vote}, \mathcal{B}, v_i]_{k_i}$ to the mixnet, encrypted with their shared key k_i.
3. The mixnet decrypts the votes, and, after mixing, sends the plaintext votes v_i to the EA.

Upon receiving the votes, the EA performs the following steps:

1. The EA checks the votes $(v_1, \cdots, v_n)$ and disregards any duplicates, posting the results of this checking step to the bulletin board.
2. The EA computes a tentative tally $\mathcal{T}'$ and publishes it to the BB.

Protocol 2. Voting.

Nullification

To nullify a vote, a hedgehog $\mathcal{H}_i$ must perform the following steps:

1. Computes a list of ciphertexts $(\mathcal{C}_1, \ldots, \mathcal{C}_n)$, each containing the encryption of 0 or 1, where each ciphertext $\mathcal{C}_j$ corresponds to voter v_j.
2. Compute a NIZKP π_j proving that each ciphertext $\mathcal{C}_j$ is the encryption of a 0 *OR* that the ciphertext is the encryption of a 1 *AND* proof of knowledge of the secret key used by voter v_i.
3. Publicly outputs the corresponding nullification pairs $((\mathcal{C}_1, \pi_1), \ldots, (\mathcal{C}_n, \pi_n))$.

Protocol 3. Nullification.

Tallying

After the voting period ends, the Tallying Authority (TA) performs the following steps:

1. The tallying authority homomorphically sums all the $\mathcal{C}_j$ for each of the hedgehogs in the system and obtains the encrypted number of nullification attempts for each voter $\mathcal{N} = \prod \mathcal{C}_j = [\sum h_j]$
2. The TA decrypts and mixes the encrypted sums and obtains the total number of votes to be nullified: h_j.
3. The TA calculates the final tally from the tentative tally by adding and subtracting the number of nullified votes determined in the nullification step, and publishes the final tally on the bulletin board.

Protocol 4. Tallying.

7 Implementation

The implementation comprises two parts. We implemented the client-side with a website similar to Helios. Voters can log in to the website to vote for some candidate. We implemented the Java website using JSP and Java Spring. Servers run on Amazon EC2 remote cloud servers. Client-side and server-side use sockets to communicate with each other, and servers also use sockets to transmit data. We use MySQL to provide database service.

We implement the Schnorr group for exponential Elgamal encryption, Schnorr non-interactive zero-knowledge proof, and Chaum-Pedersen zero-knowledge proof with 2048-bit prime number p and 256-bit prime number q. We use an inbuilt MessageDigest class provided by Java for SHA-256 hashing. The symmetric encryption is AES-256 with GCM mode by Java Cryptography Extension (JCE) library.

We implemented three interfaces for VoteXX: using an all-digital mechanism, a Remotegrity-style paper-based casting protocol pioneered by Zagórski et al. [32], and traditional polling place paper ballots with confirmation codes similar to Scantegrity II [4].

8 Conclusion

We have presented a flexible way to remove undue influence from election systems, through the use of nullification supported by voter associates whom we call hedgehogs. By separating our mechanism for mitigating undue influence from the mechanisms of ballot marking and collection, our technique works with a wide range of voting systems, including precinct voting with paper ballots, voting by mail, and Internet voting. This separation also enables engineers to build coercion-resistant voting systems that use best practices for the component parts including ballot marking and collection. Our nullification mechanism can be used in addition to other mechanisms for mitigating undue influence.

We have built a fully decentralizable VoteXX system on the xx.network, where it will be used—among other applications—for governance of the xx.network.

Having demonstrated that coercion resistance is possible, even in Internet voting, democratic societies should insist that, as a matter of due diligence, all voting systems should provide coercion resistance.

References

1. Araujo, R., Foulle, S., Traoré, J.: A practical and secure coercion-resistant scheme for Internet voting. Toward Trustworthy Elections **LNCS 6000** (2010)
2. Araujo, R., Rajeb, N.B., Robbana, R., Traoré, J., Yousfi, S.: Towards practical and secure coercion-resistant electronic elections. In: CANS (2010)
3. Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections. In: ACM STOC (1994)
4. Carback, R.T., Chaum, D., Clark, J., Conway, J., Essex, A., Hernson, P.S., Mayberry, T., Popoveniuc, S., Rivest, R.L., Shen, E., Sherman, A.T., Vora, P.L.: Scantegrity II municipal election at Takoma Park: the first E2E binding governmental election with ballot privacy. In: USENIX Security Symposium (2010)

5. Chaum, D.: Random-sample voting (2016)
6. Chaum, D., Das, D., Javani, F., Kate, A., Krasnova, A., de Ruiter, J., Sherman, A.T.: cMix: Mixing with minimal real-time asymmetric cryptographic operations. In: Gollmann, D., Miyaji, A., Kikuchi, H. (eds.) Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10355, pp. 557–578. Springer (2017)
7. Clark, J., Hengartner, U.: Selections: Internet voting with over-the-shoulder coercion-resistance. In: FC (2011)
8. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a secure voting system. In: IEEE Symposium on Security and Privacy. pp. 354–368 (2008)
9. Delaune, S., Kremer, S., Ryan, M.: Coercion-resistance and receipt-freeness in electronic voting. In: 19th IEEE Computer Security Foundations Workshop (CSFW'06). pp. 12–pp. IEEE (2006)
10. Essex, A., Clark, J., Hengartner, U.: Cobra: Toward concurrent ballot authorization for Internet voting. In: EVT/WOTE (2012)
11. Gaweł, D., Kosarzecki, M., Vora, P.L., Wu, H., Zagórski, F.: Apollo–end-to-end verifiable internet voting with recovery from vote manipulation. In: International Joint Conference on Electronic Voting. pp. 125–143. Springer (2016)
12. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: EUROCRYPT (2000)
13. Juels, A., Catalano, D., Jacobsson, M.: Coercion-resistant electronic elections. In: ACM WPES (2005)
14. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Towards Trustworthy Elections, pp. 37–63. Springer (2010)
15. Kane, C.: Voting and verifiability. Vantage Magazine **7**(1) (2010), `https://people.csail.mit.edu/rivest/pubs/Kan10.pdf`
16. Kusters, R., Truderung, T., Vogt, A.: Accountability: Definition and relationship to verifiability. In: ACM CCS (2010)
17. Lewis, S.J., Pereira, O., Teague, V.: Trapdoor commitments in the swisspost e-voting shuffle proof. `https://people.eng.unimelb.edu.au/vjteague/SwissVote.html`
18. Lewis, S.J., Pereira, O., Teague, V.: Addendum to how not to prove your election outcome. `https://people.eng.unimelb.edu.au/vjteague/HowNotToProveElectionOutcomeAddendum.pdf` (March 2019)
19. Lewis, S.J., Pereira, O., Teague, V.: How not to prove your election outcome. `https://people.eng.unimelb.edu.au/vjteague/HowNotToProveElectionOutcome.pdf` (March 2019)
20. Lueks, W., Querejeta-Azurmendi, I., Troncoso, C.: Voteagain: A scalable coercion-resistant voting system. In: 29th USENIX Security Symposium (USENIX Security 20) (2020)
21. Moran, T., Naor, M.: Receipt-free universally-verifiable voting with everlasting privacy. In: CRYPTO (2006)
22. Okamoto, T.: Receipt-Free electronic voting schemes for large scale elections. In: Workshop on Security Protocols (1997)
23. Popoveniuc, S., Kelsey, J., Regenscheid, A., Vora, P.: Performance requirements for end-to-end verifiable elections. In: Proceedings of the 2010 International conference on Electronic Voting Technology/Workshop on Trustworthy Elections. pp. 1–16 (2010)
24. Rivest, R.L.: On the notion of 'software independence'in voting systems. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **366**(1881), 3759–3767 (2008)
25. Smyth, B.: Surveying definitions of coercion resistance. (2019)
26. Specter, M.A., Koppel, J., Weitzner, D.: The ballot is busted before the blockchain: A security analysis of voatz, the first Internet voting application used in U.S. federal elections. `https://internetpolicy.mit.edu/wp-content/uploads/2020/02/SecurityAnalysisOfVoatz\_Public.pdf` (2019)
27. Spycher, O., Haenni, R., Dubuis, E.: Coercion-Resistant hybrid voting systems. In: EVOTE (2010)
28. Teague, V.: Faking an iVote decryption proof. `https://people.eng.unimelb.edu.au/vjteague/iVoteDecryptionProofCheat.pdf` (November 2019)
29. Volkamer, M., Grimm, R.: Multiple casts in online voting: Analyzing chances. In: EVOTE (2006)
30. xx.network: quantum secure true digital cash. `https://xx.network`
31. Zagórski, F., Carback, R.T., Chaum, D., Clark, J., Essex, A., Vora, P.L.: Remotegrity: Design and use of an end-to-end verifiable remote voting system. In: International Conference on Applied Cryptography and Network Security. pp. 441–457. Springer (2013)
32. Zagórski, F., Carback, R., Chaum, D., Clark, J., Essex, A., Vora, P.L.: Remotegrity: Design and use of an end-to-end verifiable remote voting system. In: ACNS (2013)