Survey of Fully Verifiable Voting Cryptoschemes

Brandon Carter, Ken Leidal, Devin Neal, Zachary Neely

Massachusetts Institute of Technology

[bcarter, kkleidal, devneal, zrneely]@mit.edu

6.857 Final Project

Abstract

Ideal voting schemes offer guarantees of voter anonymity, verifiability, and election integrity. Voting schemes used in practice today fail to meet these properties. We aim to provide context around a subset of the tools used in cryptographic voting systems and discuss a variety of cryptoschemes that have been proposed in recent literature. In this paper, we present and analyze three fully verifiable blind-signature-based voting cryptoschemes, using them as points of comparison to analyze voting schemes introduced in recent work.

1. Introduction

The design and implementation of voting systems has been widely studied in recent years. Ideal voting schemes ensure voter *anonymity*, *verifiability*, and election *integrity*.

Designing voting schemes that satisfy these properties, while also being implementable at the scale of country-wide elections, is very hard in practice. The voting system currently used for United States presidential elections falls short of attaining the properties of ideal voting schemes. For instance, the current scheme used by the United States has trouble producing convincing evidence of the outcome of an election [12]. Various end-to-end cryptographic voting schemes have been proposed to replace traditional paper- or mail-based systems, but some of these schemes rely on trusted election authorities or make other assumptions which weaken their security guarantees.

This paper aims to engineer blind-signature-based voting schemes from the ground up and use them as tools with which to analyze and critique existing schemes. We first give a formal overview of the roles in a voting system (Section 2). We then enumerate and discuss the properties required of ideal voting systems (Section 3), and analyze issues with currently used voting schemes (Section 4). In Sections 5 and 6, we critique some of the cryptographic tools relevant to designing ideal cryptographic voting schemes. In the remainder of the paper, we present three fully verifiable, blind-signature-based voting schemes and use these schemes as points of comparison to analyze schemes presented in previous work.

2. Roles

Voting schemes are often described as having three main roles: the roles of *voter*, *verifier*, and *tallier*.

2.1 Voter

A *voter* is only permitted to cast a limited number of ballots (typically only one) per election. A voter might wish that his or her ballot not be associated with him or her. In other words, a voter can vote *anonymously*. In addition, a voter would like convincing evidence *verifying* his or her vote was counted.

A corrupt voter may try to use this evidence (along with evidence of ballot ownership) to prove to another party he or she voted a certain way to receive compensation. This process is called *vote selling*, and it should be prevented, since voters may value money more than they do their right to vote. Vote selling can lead the candidate with the most funds to buy votes to win the election, breeding corruption.

2.2 Verifier

A *verifier* is responsible for verifying the eligibility of a person to be a voter.

In the case of a government election, this role is often performed by government officials. In this case, verification occurs in person using a state-issued ID. Since a verifier is responsible for deciding who is a voter, cryptoschemes often trust verifiers to perform faithfully, preventing a single voter from voting multiple times and other forms of voter fraud. In addition, though verifiers must know a prospective voter's identity, verifiers do not need to know the voter's vote (and ideally *should not* know the voter's vote; see Section 3.1).

2.3 Tallier

A *tallier* is responsible for collecting and counting the ballots and publishing the results. A tallier must be able to know whether a ballot is valid or invalid (this verification may be done through communication with verifiers or through the verification of a signature from a verifier). The tallier should provide evidence to the voter that his or her vote was counted (even if that evidence only provides a probabilistic guarantee the vote what was counted). In addition, the results from talliers should be auditable: that is, statistically verifiable by a third party.

Ideally, talliers should not know the identity of the voter who cast a ballot, even if the tallier and the verifier are the same entity.

3. Properties

The three general categories of properties desired of elections are *anonymity*, *verifiability*, and *integrity*. These properties could be satisfied given trust in an entity (in which case we consider a property *conditionally satisfied*). Otherwise, if a property is satisfied without trusting any entity, we consider the property *fully satisfied* (or, *unconditionally satisfied*). We consider the latter preferable since it provides stronger guarantees that the property is satisfied.

The difficulty in designing voting schemes can, in many cases, be attributed to the many, often conflicting properties desired of elections. For instance, if a ballot is deterministically verifiable by a voter and the voter can prove ownership of the ballot, it is possible for the voter to sell his or her vote. This issue makes remote voting difficult: it is often possible for the voter to prove ownership of the ballot and the ballot's content while outside the controlled setting of a polling booth.

3.1 Anonymity

Anonymity allows a voter to cast a ballot such that the ballot cannot be traced back to him or her. In certain schemes, it is possible to attain computational anonymity, especially with the use of blind-signatures (see Section 5.2). Some schemes achieve conditional anonymity given a trusted party (i.e. a trusted verifier, tallier, or both).

Conditional anonymity, though less strong than full anonymity, allows for other desirable properties. For example, if a verifier is trusted, then the verifier can assign a voter a unique ballot token.

*A **ballot token** is used by a voter when casting a ballot. It can be used to cast one and only one ballot. Ballot tokens and anonymity are discussed further in Section 5.1.*

If no parties are trusted with the voter's identity, no parties can know a voter's ballot token, so the ballot token must be generated randomly by the voter. If the ballot token is generated randomly, there is non-zero probability two voters will choose the same ballot token, and in this case, one user will be unable to vote. See Section 5.1 for further discussion on randomly chosen ballot tokens.

When discussing voting schemes, anonymity frequently means voting confidentiality. Anonymity in the traditional sense often implies that the anonymous parties cannot reveal their identities even if they choose to do so. With voting, however, the voter might be considered anonymous but may be able to supply a proof of his or her identity and lose that anonymity. In this sense, the voter is anonymous contingent on the fact that the voter does not reveal his or her identity. Proving identity and ownership of ballots is discussed further in Section 3.3.

3.2 Verifiability

Verifiability allows voters to verify the election results. There are two times when verifiability is desired: after a user votes, he or she would like proof his or her vote was counted (*individual verifiability*), and after the final tallies are released, many parties would like proof that the results are correct (*universal verifiability*).

These proofs can be deterministic or probabilistic. Probabilistic auditing of paper ballots provides a probabilistic proof of final vote tallies, whereas a public bulletin board of plaintext ballots provides a direct means by which anyone can deterministically verify vote tallies (see Section 5.3).

In many cryptographic voting schemes, verification is cryptographic in nature (e.g. a digital signature). However, this form of verification poses issues because each voter must either be able to perform cryptographic computations without a computer, or the voter must trust the device that performs the verification. The issue of trusting software is discussed further in Section 4.2.

3.3 Integrity

Integrity includes the properties that only eligible voters may vote, each voter may only vote once, and voters cannot sell their votes. Further, any entity should not be able to disenfranchise eligible voters for partisan reasons: by ignoring votes for a certain candidate, for example.

The process of allowing an otherwise-non-voter to vote is strictly regulated by the verifier. A corrupt verifier could easily say non-voters are eligible to vote (and even, in most cases, that a non-existent fake voter is eligible to vote). In this sense, most schemes are dependent on the verifier to only allow eligible voters to vote and to only allow these voters to vote once. For this reason, the duties of the verifier are limited to only deciding whether or not someone is eligible to vote (demonstrating the principle of least privilege).

Disenfranchisement is usually associated with the talliers. If a tallier receives a ballot voting for a candidate he or she disfavors, the tallier should not be able to ignore the ballot for this partisan reason. If the tallier ignores a ballot, guarantees of individual verifiability should make it apparent to the voter that the ballot was not counted.

In fully verifiable schemes, where plaintext ballots are often published for the public to see and verify, the ability to sell a vote is equivalent to the ability to prove ownership of a ballot. In schemes in which voters have a private key used to sign their ballot, proving ownership of a ballot could be as simple as signing a given message with the private key. In other schemes, the ability to prove ownership of a ballot could be ephemeral, usually because all the information which was once held in secret by the voter is eventually published, and afterwards the voter has no secret information to release to prove he or she voted in a certain way.

4. Voting Schemes Used in Practice

A *voting scheme* is a method of conducting an election. In our paper, we primarily focus on voting schemes in which desirable properties of an election are attained through cryptographic means, but to better understand cryptoschemes, we first explore schemes currently used in practice.

A classic paper-based voting system uses printed ballots, which are cast by voters and counted by scanners. Randomized statistical audits are performed on the ballots to support or cast doubt on the reported outcome. In this voting scheme, it is easy to guarantee at the polling location that each voter is eligible to vote and that each voter may vote only once (by only providing the voter a single ballot). Further, this scheme makes it difficult to sell votes because a voter is not given a receipt of the ballot, so the voter cannot prove which candidate was selected on the ballot. However, as a voter, there is tremendous trust placed on the election officials and auditors that votes are counted properly. The voters have no way to determine that their individual votes were counted in the election, and similarly, they are unable to recount or audit the votes themselves to confirm the outcome of the election. In a system where the government or election officials are corrupt, this voting scheme is terribly flawed due to the amount of trust placed in election officials.

In this section, we explore similar issues surrounding the current voting scheme used in the United States. Then, we examine the complexities added by having software-based components within voting systems and show how cryptographic means have been used in developing voting schemes that can facilitate anonymity, verifiability, and integrity of elections.

4.1 Issues with Voting Schemes Used in Practice

In modern United States presidential elections, each state decides how it will organize voting for its constituents. Some states rely solely on paper ballots, either at a designated polling locations (e.g. New York) or entirely by mail (e.g. Washington) [9]. Other states have adopted electronic voting systems that digitize the voting process, though voters must still vote at designated polling locations. These results can be directly recorded by polling machines and may or may not leave some form of voter verified paper audit trails that enable statistical auditing of ballots [9].

These voting schemes used during presidential elections fail to meet many of the properties laid out in Section 3. For example, consider any mail-based scheme (either for the entire set of ballots or in the case of absentee ballots). Voters sending in ballots by mail typically sign their names along with the ballot, and these ballots must be opened and processed in an election processing center. It is feasible for an employee in this processing center to record the names of voters along with their votes. Thus, the voters must trust the government for anonymity during ballot processing.

Moreover, this mail-based system is also an example of one that fails to achieve verifiability in the sense that a voter cannot verify that his or her vote was counted. An employee in the processing center could discard a particular ballot, and the corresponding voter who sent that ballot would never be able to tell whether the vote was counted in the election.

The final counts of the election are probabilistically verifiable via audits. However, voters are unable to perform this verification themselves, and instead must trust a third party to audit.

Mail-based voting schemes further lack the desired integrity of successful voting schemes. These schemes make vote-selling easy. For example, a constituent can easily sell his or her vote by filling out and mailing the ballot in the presence of the buyer, who is then guaranteed knowledge of how the ballot was filled out.

It is clear than that the voting schemes currently used in United States presidential elections are insufficient at attaining the properties desired of an ideal voting scheme. In response, many cryptographic voting schemes have been proposed, in particular to address the issues surrounding anonymity, verifiability, and integrity of elections. However, these cryptographic voting schemes are implemented in software in practice, which introduces other concerns regarding trust in the software and hardware.

4.2 Issues with Software-Based Schemes

Recently, software-based systems have been adopted to help simplify the voting and counting processes. However, electronic voting systems can contain any number of issues ranging from software bugs to malicious code that can determine an election regardless of cast votes [10].

A fundamental issue with software-based schemes is that voters must trust that the software was both well-written and well-tested and that the software running on polling machines is authentic [10]. Since large software systems are often very complex, attaining the properties of ideal voting schemes in a purely software-based solution is quite difficult.

In 2006, Rivest and Wack proposed a notion of *software-independent* voting systems, in which an undetected error in the software cannot cause an undetectable change in the election outcome [10]. A system is said to be *strongly software-independent* if it is possible to correct any error in the outcome.

These notions of software independence are generally necessary in order to design an ideal voting scheme which has software components. Though we do not focus on system implementation in this paper, it is important to understand the notion of software independence in the implementation of voting cryptoschemes.

5. Tools

Before we present specific voting cryptoschemes, we first introduce several tools that comprise these systems. After discussing these tools, we treat them as black-boxes that allow us to simplify the description and analysis of voting schemes.

5.1 Random Ballot Token

In order to achieve anonymity, a voter, Alice, needs a secret ballot token which proves she is eligible to vote once (and only once), but which cannot be directly associated with her. To attain this level of anonymity in a scheme in which ballots are published, the ballot token cannot be assigned by the the verifier because then the verifier would know Alice was assigned a specific ballot token. For that reason, in many of our schemes, Alice generates this token randomly. As a consequence, however, it is possible (but highly unlikely) that two voters may randomly select the same ballot token.

In many of the schemes we explore, in the event of a ballot token collision, whichever voter cast a ballot first would have his or her vote counted, and the other voter would not be able to vote, since a token can only be used to cast one vote. In theory, with a large enough ballot token length and a truly random process for selecting tokens, the probability two voters choose the same secret is negligible, but it should be a concern if using such a strategy in practice.

In the first two schemes we present, the ballot token is a keypair. In the third scheme, it is a random number.

5.2 Blind Signatures

Blind signatures allow verifiers to sign a secret held by a voter without knowing the secret. This method is particularly useful for verifiers to validate a voter's ballot or ballot token without knowing it explicitly. For example, blind-signing a ballot token could work as follows: a voter generates a random ballot token (see Section 5.1) and sends a proof of his or her identity and a blind-signing request for the ballot token to a verifier. The verifier then verifies the proof of the voter's identity and blind-signs the blind-signing request and returns it to the voter. From the response, the voter can extract the signed ballot token.

The blind signing procedure must *hide* the secret from the signer. Blind signatures must also be *unforgeable*, meaning signatures for any other secret cannot be extracted from the returned signature.

In practice, a blind-signature scheme derived from RSA may be used for blind signatures [4, p.235]. If Alice wants Bob to sign m without telling Bob what m is, she can send Bob $m \cdot r^e \mod N$ (where r is chosen randomly from $\mathbb{Z}_N$ and e is Bob's public exponent in the RSA scheme), Bob can exponentiate by d (the private exponent) and send the result to Alice, and Alice can divide by r to attain $m^d \mod N$, a valid signature for m. A diagram illustrating the blind signing process is given in Figure 1.

It is important to note, however, by exponentiating $(m \cdot r^e)^d \mod N$, Bob is signing $m \cdot r^e$, which means that Alice has two signatures from Bob: one for m and one for $m \cdot r^e$. If Alice is only allowed to have a signature for one message, this issue can be prevented if $h(m)$ is blind-signed instead of m where h is a one-way, collision resistant hash function, because it would be computationally difficult for the voter to find a second message, m', such that $h(m') = h(m) \cdot r^e$. Verifying the signature would then involve hashing m with h and exponentiating the signature ($h(m)^d$) by e to see if the same value is attained. In addition, Bob must *only* use the blind-signing RSA key for blind-signing, since signing RSA is equivalent to decryption, meaning Bob acts as a decryption oracle.

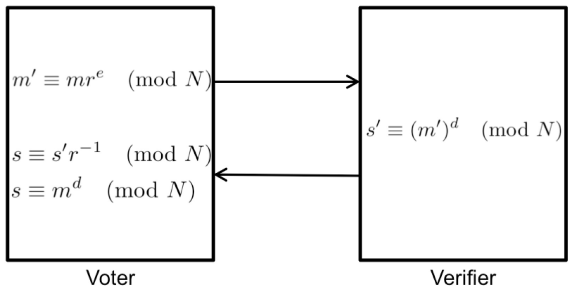

Figure 1: *Blind signing m* [1]

[1]Based on a figure at https://it.wikipedia.org/wiki/Blind_signature

5.3 Public Bulletin Boards (PBB)

The goal of using *public bulletin boards (PBBs)* is to attain public verifiability by publishing information in the public sphere. In voting schemes, this published information often takes the form of ballots, which may be plaintext or ciphertext from a homomorphic encryption scheme. PBBs might be implemented as a publicly maintained blockchain, a peer-to-peer network, or a signed list published on a centralized server and mirrored by parties to ensure records are not removed. When implementing a PBB, one must be concerned with the altering of the PBB (i.e. removing entries or adding false entries to the PBB). In addition, since the PBB is more reliable if it is either publicly maintained or publicly mirrored, a system should consider how and why participants choose to participate in maintaining/mirroring the PBB.

5.3.1 Centralized

A centralized PBB might be implemented as a public HTTP server maintained by a tallier. The tallier might sign each entry on the PBB to prove it has been published on the PBB or sign the root of a Merkle tree of entries. This protocol allows third parties to mirror the PBB to ensure that the verifier cannot disenfranchise voters by removing votes which were published at the time the PBB was mirrored.

In certain voting schemes, the proof given to the voter that his or her ballot was counted is the fact that the ballot is published on the PBB. In these schemes, it is necessary that the voter mirror the PBB (or, if each entry in the PBB is signed, at least the signed entry which proves the voter's ballot was published on the PBB) to ensure that the tallier cannot later remove the voter's ballot from the PBB.

5.3.2 Peer-To-Peer (P2P)

One negative aspect of using a centralized server to publish the PBB is that the server is a central point of failure and an easy target for distributed denial of service (DDoS) attacks. This problem warrants exploration of peer-to-peer methods of maintaining the PBB.

Peer-to-peer networks can be used to maintain the PBB in a distributed matter. Multiple PBBs could be maintained by peers in the network where each peer is a tallier. When a voter tries to verify his vote will be counted, the more peers which have accepted the ballot into their PBB, the more likely it is the ballot will persist to be counted in the final tally.

Eventual consistency can be driven if there is a competitive element introduced to the collection of ballots on peers' PBBs: a reward is split among all peers which have the most number of verified ballots published. This reward incentivizes peers to publish as many ballots as possible and prevents peers from ignoring ballots for partisan reasons. Rewards can be a problem, however, if higher rewards are offered by a malicious party to peers which ignore certain ballots for partisan reasons, but then there will ideally be a non-partisan peer which publishes *all* ballots and receives a higher reward than those peers ignoring votes for partisan reasons. In addition, peers might hide votes published on their PBB until very late such that they do not share ballots with competing peers. This game theory problem of whether to hide ballots or share them complicates matters, showing how difficult it can be to regulate competition when incentives are introduced to drive a P2P network. If incentives are introduced, one way to manage competition is through a blockchain.

5.3.3 Blockchain

Blockchains can be used to maintain a PBB by incentivizing miners to mine blocks consisting of a list of valid ballots, just like blocks consisting of lists of valid transactions are mined in Bitcoin. However, with voting, there is the added difficulty in that miners can be incentivized by a partisan third party to only mine transactions consisting of ballots for a certain candidate. In this situation, less powerful, non-partisan miners would ideally have more time to mine blocks of ballots ignored by partisan miners at the end of the election when the number of yet-to-be-mined ballots are dwindling. This process could, however, require a voter to wait a long time before his or her ballot is accepted into the blockchain.

5.4 Commitments

Cryptographic *commitment schemes* provide beneficial properties for integrity. Most notably, commitments can be used to help postpone disenfranchisement for partisan reasons (but not prevent it, as will be shown). For example, if a voter submits a commitment to a vote, the tallier can sign the commitment, promising to count the vote before the voter reveals the vote. This procedure holds the tallier accountable to counting the vote in the final tally. If the ballot were *not* hidden by the commitment, the tallier might ignore the vote for partisan reasons and claim the vote was never sent. The commitment strategy is used in the voting scheme described in Section 6.1. The one problem with this approach is that the potential for disenfranchisement is deferred to the reveal. If a voter attempts to reveal his or her ballot commitment, the party maintaining the PBB of published revelations can ignore the revelation (and claim it was never sent).

5.5 Mixnets

To attain full voter anonymity, voters must be able to send ballots to talliers without the tallier being able to track the ballot back to the voter through the transmission medium. *Mixnets* solve this problem by taking in these ballots from voters, shuffling them (while keeping the permutation secret), and then sending the shuffled ballots to the talliers.

Many varieties of mixnets have been proposed, though a class known as *robust mixnets* is most useful for applications such as voting because these mixnets have strict correctness requirements (i.e. ballots cannot be lost or corrupted during shuffling) without time-sensitivity constraints (it is okay if the mixnet collects ballots over hours or days before performing the shuffling) [1, p.73]. Further, these mixnets can make provable guarantees about the privacy of the shuffle permutation.

An extensive review of verifiable mixnets is given by Adida [1, Ch.3]. For the purposes of our paper, we assume that mixnets allow us anonymous one-way transport of ballots from voters to talliers, similar to how they are used in a voting scheme designed by Sako and Kilian [11]. We further assume mixnets can make guarantees of universal verifiability of correct mixing and voter privacy.

6. Fully Verifiable Voting Schemes

In beginning to explore voting cryptoschemes, we started with the goal of designing a voting scheme which is fully verifiable and which allows the voter complete anonymity. This goal naturally led to the exploration of the use of random ballot tokens, blind-signatures for verification, and public bulletin boards for publishing plaintext ballots.

The following voting schemes are presented in the chronological order in which we explored them, and each scheme is a simplification of the previous. While designing the schemes, we learned that designing cryptoschemes often involves a cycle of adding features and then stripping away other features which add complexity without necessarily adding desirable properties.

6.1 Register-Commit-Reveal (RCR) Scheme

Our first attempt at a fully verifiable voting scheme is called the *register-commit-reveal (RCR) scheme*. The RCR scheme is divided into three phases: registration, commitment, and reveal. During registration, the voter, Alice, generates a keypair for an assymetric encryption scheme. The public key of this keypair is Alice's ballot token. Alice submits a proof of her eligibility to vote (e.g. government-issued ID) and a request for a blind signature on her public key to the verifier. The verifier has a blind-signing key which is only used for blind-signing public keys for the current election. The verifier checks to ensure Alice has not previously requested a blind-signature for the election and that Alice is eligible to vote. If so, the verifier blind-signs Alice's public key and sends the blind signature to Alice.

During commitment, Alice uses a cryptographic commitment scheme (ideally hiding and binding) to commit to a ballot and generates a signature for the commitment using her private key. Alice sends her public key, the verifier's signature of her public key, the commitment, and the signature for the commitment to the verifier via a mixnet (to avoid tracking the ballot to Alice via the network). The verifier verifies that the signature for Alice's public key is valid, verifies Alice's signature for the commitment is valid, and then signs the commitment and publishes the signed commitment to a public bulletin board. This signature is the verifier's validation of the legitimacy of the ballot and is made without the verifier knowing the ballot's content or to whom the ballot belongs. After the public bulletin board is finalized, the reveal phase begins.

In the reveal phase, each voter checks to ensure his or her commitment was signed by the verifier and placed on the public bulletin board. Then, voters send the information to reveal their ballots to multiple third party talliers (e.g. Wikileaks[2]), again via mixnets. These talliers verify that the information reveals commitments signed by the verifier (by consulting the public bulletin board of verifier signed commitments and verifying the ballot is what the voter committed to during the commitment phase) and signs and publishes valid revelations on their own public bulletin boards. It is then possible for anyone to view and count revealed plaintext ballots.

6.1.1 Properties

For a voter, Alice, this scheme is anonymous given anonymous transport (via mixnets), so long as Alice chooses not to reveal her identity. If, however, Alice chooses to reveal her private key at any point (or interactively sign a message with her private key), it is possible for her to reveal her identity. The scheme is also deterministically verifiable: since the ballots are published publicly, anyone can verify the final tally (universal verifiability), and any voter can verify his or her vote was included in the public bulletin board (individual verifiability) by keeping a tallier-signed-and-published receipt of the revelation. These two properties combined, however, lead to the undesirable property that it is easy for Alice to sell her vote simply by proving ownership of her private key after her plaintext ballot has been published on a public bulletin board. Moreover, if there is time between the registration and commitment phases, Alice can simply sell her private key and verifier-signed public key to a malicious third party and the malicious third party can commit and reveal a ballot for Alice.

Though vote selling is possible, the RCR scheme has other desirable properties, such as unforgability. Votes are unforgeable given (1) a trusted verifier (who will only blind-sign for eligible voters and only once per voter), (2) a binding commitment scheme (so a malicious third party cannot reveal a different ballot for a published commitment), (3) unforgeable signatures. In addition, though registration may be performed in person (where it is easier to verify a voter's identity) commitment and reveal can be performed remotely, adding a layer of physical anonymity as well as convenience for the voter.

[2]https://wikileaks.org/

Cryptographic commitments provide the desirable property that the receiver cannot ignore a commitment to a ballot for partisan reasons, since the commitment hides the ballot. However, commitments merely add complexity to the scheme without eliminating the potential for disenfranchisement, since the party which receives the revelation of the commitment can then ignore the revelation for partisan reasons. In this system, since there are multiple third party talliers, so long as one of the talliers accepts the revelation of the commitment, the ballot is counted. However, the use of commitments adds one layer of complexity while still allowing a group of talliers to collude to ignore a voter's ballot revelation, disenfranchising the voter. For this reason, we remove the level of indirection introduced by commitment in our next scheme.

6.2 Register-Vote (RV) Scheme

The *register-vote (RV) scheme* is a simplification of the register-commit-reveal (RCR) scheme which removes the unnecessary complexity of commitments. In the RV scheme, the registration phase is identical to the registration phase of the RCR scheme (see Section 6.1). In the vote phase, a voter, Alice, signs her ballot with her private key and sends the signed ballot, her public key, and the verifier's signature of her public key to one or more talliers. The talliers verify the legitimacy of the verifier's signature of Alice's public key and Alice's signature of the ballot. If both verifications pass, the tallier signs and publishes the received information on the tallier's public bulletin board.

6.2.1 Properties

The RV scheme has the same properties as the RCR scheme:

1. As long as one tallier accepts and publishes Alice's vote, it is counted, but if the talliers collude, they can disenfranchise a voter by ignoring his or her vote.

2. Since the ballots are posted publicly, anyone can verify the final tallies (universal verifiability).

3. Since the ballots are posted publicly and signed by talliers, voter's have proof their ballots were counted (individual verifiability).

4. Votes are unforgeable given a trusted verifier.

5. Alice is anonymous given anonymous transport, but can reveal her identity if she wishes by proving ownership of her private key.

6. Votes can be sold in the same manner as in the RCR-Scheme.

without the unnecessary level of complexity introduced by commitments. However, this scheme can be simplified further. If registration and voting occur at the same time (e.g. in a polling location), they can be performed in the same step.

6.3 Verify-Vote (VV) Scheme

The *verify-vote (VV) scheme* is a simplification of the register-vote (RV) scheme. In the VV scheme, a voter, Alice, generates a random ballot token. Alice then sends proof of her identity and a blind-signing request for her ballot token concatenated with her ballot to the the verifier, who verifies Alice's identity and blind-signs her ballot token concatenated with her ballot. Alice then sends her ballot token, her ballot, and the verifier's signature of her ballot token and ballot to one or more talliers, who verify the verifier's signature. The talliers then sign and publish the information on their PBBs.

6.3.1 Properties

The VV scheme has all the desirable properties of the RV scheme, but it has the additional desirable property that once Alice submits her vote, she can no longer prove her ownership of a ballot, since all secret information she possessed is in the public sphere[3]. As a result, Alice can only sell her vote in the time between the verifier's blind-signing of her ballot token and the time her vote is published on a tallier's public bulletin board (i.e. when her secrets become public knowledge). This ephermality provides a stronger guarantee of voting integrity than in the RV or RCR systems. At the same time, by combining the steps, we decrease the convenience for the voter, since the voter's ballot must be decided at the time of registration, meaning registration cannot be performed beforehand and the vote decided later.

7. Analysis of Previous Work

In the following section, we analyze four previously proposed voting schemes and use the blind-signature-based schemes we explored as points of comparison. In addition, Table 1 outlines the basic properties of our schemes and the previously-published schemes that we analyze.

A paper by Ibrahim et al. is often cited as one of the first papers to explore the use of blind signatures in electronic voting schemes [5]. In the E-Voting System (EVS) proposed by Ibrahim et al., the verifier assigns each voter a ballot token during registration. During voting, the verifier blind signs the voter's ballot. The voter then sends the signed ballot to the tallier, who sends a signed receipt of receiving the ballot to the voter. The tallier publishes the plaintext ballots on a public bulletin board.

Though our proposed VV scheme is similar to Ibrahim et al.'s EVS scheme [5], the VV scheme offers better guarantees of anonymity and makes vote selling more difficult. In the EVS scheme, since the verifier assigns the voter a unique ballot token and the ballots are published on the tallier's public bulletin board, the verifier can map the ballots back to the voters who cast them. As a result, the EVS

[3]that is, unless Alice sells to the verifier by revealing her blind-signing secret, but the verifier has the power to counterfeit ballots anyway, so doing so would be pointless

Name	**Anonymous**	**Proof Owns Ballot**	**Unique IDs**	**Assumptions**
RCR	Given anon one-way transport	Yes, sign with SK	Probabilistic	Random keys, strong RSA, OW/CR hash, binding/hiding commitments
RV	Given anon one-way transport	Yes, sign with SK	Probabilistic	Random keys, strong RSA, OW/CR hash
VV	Given anon one-way transport	Ephemeral	Probabilistic	Random IDs, strong RSA, OW/CR hash
Ibrahim et al. [5]	Given trusted verifier	Yes, receipt selling	Deterministic	strong RSA, OW/CR hash
HandiVote [8]	Given trusted election authority and anon transport, vote history between elections	Yes, sell card/pin	Deterministic	4 digit PIN, SMS, phone, and ATMs are secure
Kucharczyk single-election [6]	Given anon one-way transport	Ephemeral	Unspecified	strong RSA, OW/CR hash
Kucharczyk multi-election [6]	Given anon one-way transport, vote history between elections	Yes, sign with SK	Probabilistic	Random keys, strong RSA, OW/CR hash
Fujioka et al. [3]	Given anon one-way transport	Ephemeral	Probabilistic	Random IDs, strong RSA, OW/CR hash, binding/hiding commitments

Table 1: *Side-by-side comparison of properties of fully verifiable voting schemes.*

scheme's anonymity is conditional on trust in the verifier. The fact that the verifier assigns unique ballot tokens does, however, ensure there can be no ballot token collisions, which is not the case in our schemes (in which a voter randomly chooses a ballot token). This fact highlights a tradeoff between full anonymity/verifiability and deterministically unique ballot IDs. Probabalistically unique ballot tokens are so unlikely to collide that we feel the tradeoff for stronger anonymity guarantees is acceptable.

In addition, in Ibrahim's EVS, the tallier has an active session with the voter and engages in request/response interaction, which means that transport from the voter to the tallier cannot be anonymous and one-way, which again compromises anonymity.

Finally, because the tallier in EVS returns a signed receipt of receiving a ballot, this receipt serves as a weak proof of ballot ownership which could be sold by the voter as a means of vote selling, even if voting were performed in a controlled environment, like a polling booth. In contrast, our VV scheme only gives ephemeral proof of vote ownership until the point that the ballots and tallier-signed receipts are published (since everyone has access to all tallier-signed ballots after they are published).

In a paper by Kucharczyk, a voting cryptoscheme is proposed which is very similar to VV, except that it is not specified that ballot tokens must be generated randomly to achieve full anonymity [6, p.356]. Kucharczyk claims that such a scheme would be too inconvenient for a voter, since a new blind-signature would be needed for each election. He instead proposes a scheme similar to RV in which a public key is blind-signed by the election authority and used by the voter to vote in multiple elections. However, as noted in the analysis of RV, this process allows for easy vote selling through the selling of the private key and signed public key. Vote selling is particularly dangerous since the key might be used for multiple elections. In addition, though it might not be possible to link the key to the voter, it is possible to see the voting history for a single key across multiple elections. This metadata compromises anonymity, to a degree.

In a similar vein as RV and Kucharczyk's proposed system, a paper by Renaud and Cockshott proposes a system called HandiVote [8]. In HandiVote, voters are issued securely sealed envelopes containing a voter card with an associated PIN. They use this voter card and PIN to cast ballots remotely for multiple elections, and plaintext ballots (with associated voter card numbers) are published after the election [8]. However, being assigned a "random" card has poor anonymity guarantees: the voter must trust that the issuing authority has not written down the card number and mapped it to the voter. In addition, there can be major problems if the card and PIN are lost, stolen, or sold (it is easy to sell votes—even for multiple elections—by selling the card and PIN). In addition, it is easy to identify the voting history of a card number, even if the name of the voter is not linked to the card number. In addition, it is possible for a voter to be disenfranchised by the election authority if the election authority claims that the PIN the voter used to vote is incorrect and the card is counterfeit: no one can prove that the card was, in fact, issued by the election authority. In short, HandiVote heavily relies upon a trusted election authority and does little to protect against vote selling.

A scheme proposed by Fujioka et al. closely resembles our VV scheme, but instead of requesting the verifier blind-sign a vote, the voter requests the verifier blind-sign a binding and hiding commitment to a vote (similar to the commitment used during the commit phase of our RCR scheme) [3]. The authors claim that by using a commitment, if the verifier refuses to blind-sign the ballot or the tallier refuses to publish the signed ballot, the voter can claim the verifier/tallier violated the protocol without revealing the voter's vote. Although the voter does not have to reveal the vote if the protocol violation occurs during commitment, if the violation occurs during the reveal, the voter will have to reveal his or her vote to claim a violation occurred. That is, using the commitment gains the scheme little benefit over simply publishing the vote. The commitment merely defers the problem to the reveal stage (as we found for RCR). In addition, by adding the extra step of revelation, the voter is given another window of opportunity during which he can sell his or her vote (by selling the information needed to reveal the ballot). Therefore, we find that the simplified VV scheme still upholds the same security guarantees as Fujioka et al.'s scheme despite not using commitment to initially hide the vote from talliers [3].

8. Discussion

Although the schemes we presented and evaluated make much more powerful guarantees about the election process, the additional overhead required to implement them securely is another challenge to be addressed before they can be used in practice. The biggest barrier in this respect is the fact that it is much more difficult to prove to a voter that a machine the voter has never seen before does in fact do what it claims to do. Software is usually the weakest point in any cryptographic implementation, which makes it unfavorable to place trust in any machine responsible for tallying, encrypting, publishing votes, etc. (as discussed in Section 4.2). The complexity and precision required to ensure end-to-end voting is secure makes it difficult to use practically.

However, it is worth noting that a system in which each vote is posted publicly goes a long way toward solving this problem, since in theory anyone can verify that the reported counts are correct (by hand if necessary). The first and most important step in bringing end-to-end voting into the public domain is realizing the egregious problems in our current voting system, and understanding the need to fix them. Making progress toward the development of cryptographic means of voting will ultimately help prevent corruption in this quintessential aspect of democracy: verifiable elections.

9. Conclusion

In this paper, we discuss the complexities involved in designing cryptographic voting schemes that attain various degrees of anonymity, verifiability, and integrity. We describe some of the tools that comprise these schemes, such as blind signatures, commitments, and mixnets.

We present and analyze three voting schemes which are fully verifiable. Our schemes also attempt to make guarantees about voter anonymity and election integrity. In particular, our schemes extend blind-signature-based voting schemes.

Our first scheme, RCR, allows for both universal and personal verifiability as well as ballot unforgeability. However, the RCR scheme suffers from its inability to prevent vote selling and complexity.

Our second scheme, RV, is a simplification of the RCR scheme because it removes complexity introduced by the commitment phase. In addition to being simpler, the RV schemes has the same properties as the RCR scheme.

Our final scheme, VV, further simplifies our model by combining the registration and vote phases. It satisfies the same properties as the RCR and RV schemes, and additionally, it makes vote selling harder because votes can only be sold for the short period of time between a ballot being blind-signed and being published.

We compare our schemes to previously proposed blind-signature-based schemes. We claim that the schemes we present leverage simplicity with the cryptographic properties desired of successful voting schemes.

Acknowledgments

We would like to thank our professor Ron Rivest and project advisor Conner Fromknecht for their valuable input on early iterations of our cryptoschemes.

References

[1] Adida, B. (2006). Advances in Cryptographic Voting Systems. PhD thesis, MIT Department of EECS. http://crypto.csail.mit.edu/ cis/theses/adida-phd.pdf

[2] Benaloh, J. (1994). Dense Probabilistic Encryption. Workshop on Selected Areas of Cryptography. http://research.microsoft.com/en-us/um/people/benaloh/papers/dpe.ps

[3] Fujioka A., Okamoto T., and Ohta, K. (1992). A practical secret voting scheme for large scale elections. *Advances in Cryptology: Lecture Notes in Computer Science*, 718, 244-251. doi:10.1007/3-540-57220-1_66

[4] Goldwasser, S. and Bellare, M. "Lecture Notes on Cryptography". Summer course on cryptography, MIT, 1996-2001. http://cseweb.ucsd.edu/ mihir/papers/gb.pdf

[5] Ibrahim, S., Kamat, M., Salleh, M., & Aziz, S. R. A. (2003). Secure E-voting with blind signature. In 4th National Conference on Telecommunication Technology Proceedings. NCTT 2003. (pp. 193-197). IEEE

[6] Kucharczyk, M. (2010). Blind Signatures in Electronic Voting Systems. *Communications in Computer and Information Science*, 79, 349-358.

[7] Pallier, P. (1999). Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. http://www.cs.tau.ac.il/ fiat/crypt07/papers/Pai99pai.pdf

[8] Renaud, K. and Cockshott, P. (2009). HandiVote: Simple, Anonymous, and Auditable Electronic Voting. *Journal of Information Technology & Politics*, 6, 60-80. doi:10.1080/19331680802668102

[9] Rivest, R. L. "Auditability and Verifiability of Elections". ACM-IEEE talk, 2016. http://courses.csail.mit.edu/6.857/2016/files/L20-Auditability-and-Verifiablity-of-Elections-slides.pdf

[10] Rivest, R.L. and Wack, J.P. (2006). On the notion of "software independence" in voting systems. https://people.csail.mit.edu/rivest/pubs/RW06.pdf

[11] Sako, K. and Kilian, J. (1995). Receipt-Free Mix-Type Voting Scheme: A practical solution to the implementation of a voting booth. *Advances in Cryptology: Lecture Notes in Computer Science*, 921, 393-403. doi:10.1007/3-540-49264-X_32

[12] Stark, P.B. and Wagner, D. (2012). Evidence-based Elections. *IEEE Security & Privacy*, 10(5), 33-41. doi:10.1109/MSP.2012.62