Volume 4, Issue 5, May 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Evolution in Cryptographic Voting System

Shridhar
Research Scholar
Department of Computer Science
Maghad University
Bodh Gaya (Bihar)

Abstract: ***This present recent advances in cryptographic voting systems, a type of election system that provides mathematical proofs of the results—rather than of the machines. We begin with a review of the functional requirements of voting, the specific issues that makes voting fairly complex, and an overview of the basic technical concepts behind cryptographic voting. We note with interest that, to this day there is no known way to execute a truly secure and verifiable election without some elements of cryptography.***

Keywords: Cryptograph,Cryptographic,Voting system.

I. INTRODUCTION

One of the most important tasks of a democratic government is the planning and execution of the election that designates its successor. Not surprisingly, it is also one of its most challenging tasks, one whose requirements and constraints are remarkable strict. Thus, dubious results, failing technology, and ingenious methods of fraud have been noted throughout election history. Lever machine counters have been rigged, ballot boxes have been lost or magically found and dead citizens have voted. Legitimate voters have been coerced in various ingenious ways, including chain voting, spreading false election date location information, and instilling false fears regarding the consequences of showing up to vote (arrests, jury duty, …)

In the United States, the 2000 Presidential Election caused much stir: Bush lost the popular vote but won the Electoral College, including a win in Florida by a margin of 500 votes [67]. Numerous complaints were aired: the "butterfly ballot" in Broward Country was misleading, the punchcard systems failed to record a number of votes, and more than 50,000 absentee ballots went missing [10]. This debacle served as a public wake-up call that elections are far from perfect.

Equipment failures were well known to election officials long before Gore vs Bush [9]. However, these failures had not previously been implicated in such a close-call election. The fallout from the drawn-out results-certification process of 2000 caused many states to reconsider their voting equipment, often including hastened moves to newer, more computerized solutions so as to be "not like Florida" [113]. These changes raised questions of their own, notably among a number of computer scientists who feared that fully computerized voting would complicated or completely prevent the election verification process [11, 163]. The controversy and debate are ongoing, with many discussions about what will happen in the upcoming 2006 mid-term elections.

II. Classic Voting Systems

In the conflict between auditability and secrecy, election systems must often favor one or the other in a compromise. All election systems used in the United States since the introduction of the secret ballot in 1892 have favored secrecy over audit ability. Typically, secrecy is ensured by forced physical dissociation of identity and ballot, e.g. dropping an anonymized ballot in a physical ballot box. On the other hand, election auditing generally depends on a properly enforced chain of custody of election equipment and ballots.

A. Chain - of – Custody Security

Current voting solutions use some type of voting machine to assist voters in preparing and casting a ballot. The machines are built by private companies, according to various state- specific standards. Independent testing agencies (ITAs) are called upon to evaluate and certify the various voting equipment, though their lack of power and limited means has often been criticized [51]. Election officials generally perform additional tests, usually in the weeks prior to election, and, sometimes, in parallel with the election [58], to provide assurance that the machines are working as expected. On election day, voters cast their ballots via these machines, and resulting votes end up in some ballot box, either physical or digital. Election officials then transport these ballot boxes to a counting facility, where they are tallied. The aggregate results are finally posted for all to see. This approach presents three major limitations:

1. **Voters must verify by proxy:** only election officials can ensure that various testing procedure are adequate to begin with. Voters receive some indirect indication of tests results, such as the machine testing receipts posted on the wall at the start and end of election day. However, voters cannot directly verify that the ballots boxes themselves (digital or physical) are handled appropriately throughout election day.
2. **Verification strongly depends on chain-of-custody:** Election officials must maintain a well-audited chain-of-custody to defend against malicious problems. For an election to run correctly, every transition must be performed correctly, and the chain of custody of machines and ballots must be strictly respected at all times. A single failure can open the door to significant corruption of the election results.
3. **Recovery is very difficult:** the error detection mechanism are few coarse- grain, able only to notice honest mistakes, rarely malicious attacks. If an error is detected, some voting systems allow for ballots to be recounted. Such a recount can only address failures in the tallying process, however recovering from an integrity breach on ballot boxes or voter intent capture requires that the election be re-run, as it is usually impossible to tell the legitimate ballots from the fraudulent ones, or properly cast votes from improperly cast votes.
   In other words, in order to implement the secret ballot, current voting systems resort to a trust-by-proxy, chain-of-custody-based verification mechanism, with a "point-of-no-return" ballot hand-off, beyond which recovery is limited. This approach is both highly constraining and prone to significant error. Whether paper, optical-scan, or touch-screen, classic election methods to significantly compromise verifiability in order to achieve ballot secrecy. This chain-of-custody verification process is diagrammed in Figure 1-2.

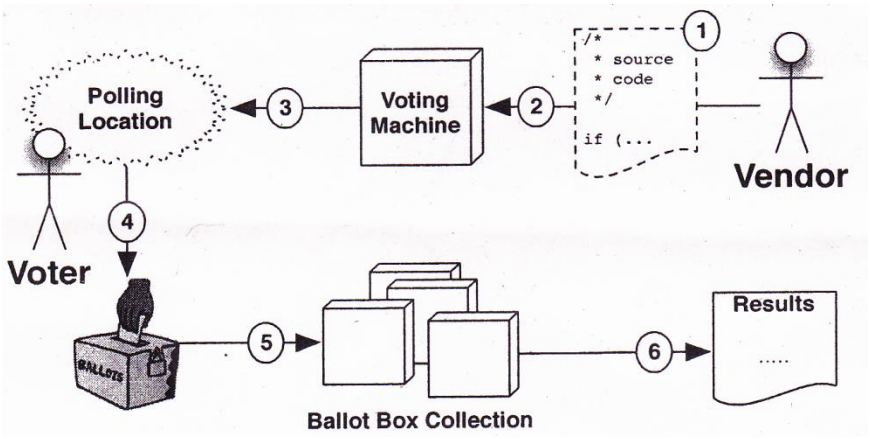


Figure 1-2: Chain-of-Custody Voting

B. The Erosion of the Secret Ballot

As a results, a number of new election proposals have tipped the scales in the other direction: in order to address the lack of verifiability in current election systems, they propose to every step must be verified. (1) The source code for voting machines is read and checked. (2) The installation of the voting machine is verified to ensure that the verified software is indeed installed. (3) The voting machines are sequestered and sealed prior to the election, and must be secured against physical attacks (e.g. installation of malicious hardware components). (4) Election officials ensure that only eligible voters cast a ballot. (5) Ballot boxes are scaled and collected with care. (6) Tallying occurs in a secured area, ensuring that no ballots are maliciously lost or injected.

compromise ballot secrecy. In some cases, this erosion of ballot secrecy occirs without anyone noticing. There is, in some election communities, a surprising belief that election integrity can be achieved even if the system is coercible

Internet Voting (Switzerland). Switzerland holds multiple referenda per year. In order to improve convenience and increase voter turnout, the Swiss have chosen to introduce Internet-based voting using any home computer [35]. Unfortunately, internet voting is inherently coercible [50], as there is no guaranteed privacy during ballot casting.

Vote By Mail (Oregon and Beyond). Oregon has long offered voting by mail to its residents, in part due to the large distances voters have to travel to reach a polling location. Recently, a grassroots effort has emerged to promote the same vote-by-mail approach in other states [172]. This group strives to make elections more convenient and bypass the complexity and cost of in-person elections. Most recently, a task force in San Diego suggested a move to permanent vote-by-mail for all [153]. Unfortunately, much like Internet voting, vote-by-mail is inherently susceptible to coercion, with detection almost impossible.

Return to Public Voting? Among the voting activism groups, some individuals have recently proposed explicitly doing away with the secret ballot altogether. They claim that "auditability and secrecy are incompatible" [112], propose that auditability is more important than secrecy, and conclude that we should simply give up ballot secrecy.

Unfortunately, though the secret ballot may seem inconsequential, it is, historically, one of the more important developments in democratic elections. A recent study of Chilean election data shows that ballot secrecy, introduced for the

first time in Chile's 1958 elections, has "first order implications" on the election results and subsequent policy decisions [97]. Coercion is notoriously difficult to detect. and, though not currently rampant, may return in force if the secret ballot is compromised. We have reached a critical juncture, where the public outcry for verifiability must be addressed, for fear that the pendulum will swing too far in the other direction, jeopardizing the secret ballot and thus the ability to honestly gauge voter intent.

C. The Voter-Verified Paper Audit Trail

As previously described, one proposed solution for verifiability is the Voter-Verified Paper Audit Trail, abbreviated VVPAT. Though there are some concerns regarding the practicality of VVPAT machines [166], there is no question that a properly operating VVPAT machine would significantly simplify the verification chain. VVPAT effectively short-circuits the voting equipment: voters get the ease-of-use associated with computer voting, while the paper trail provides a direct mechanism for verification of the voting machine's output.

However, it is worth noting that even VVPAT does not change the *nature* of the verification process: a chain of custody, albeit a shorter one, must still be maintained and audited, so that the following questions can be answered:

- Do the accepted paper trails get properly deposited in the ballot box? Do the rejected paper trails get properly discarded?
- Are the ballot boxes of paper trails appropriately safeguarded during election day?
- Are the ballot boxes of paper trails appropriately collected and safeguarded after the polls close? What are the safeguards against the introduction of extraneous ballot boxes?
- Are the paper trails properly tallied? Using what process?
- In other words, the VVPAT short-circuits the custody chain prior to ballot casting. The verification process for everything that follows the ballot hand-off, however, remains a chain of custody that must be properly enforced at all times.

III. Cryptographic Voting Schemes

We now consider cryptographic voting systems at a high level. We pay specific attention to the end-to-end nature of the verification provided by such systems, and how, consequently, any observer can verify the proper operation of a cryptographic election completely.

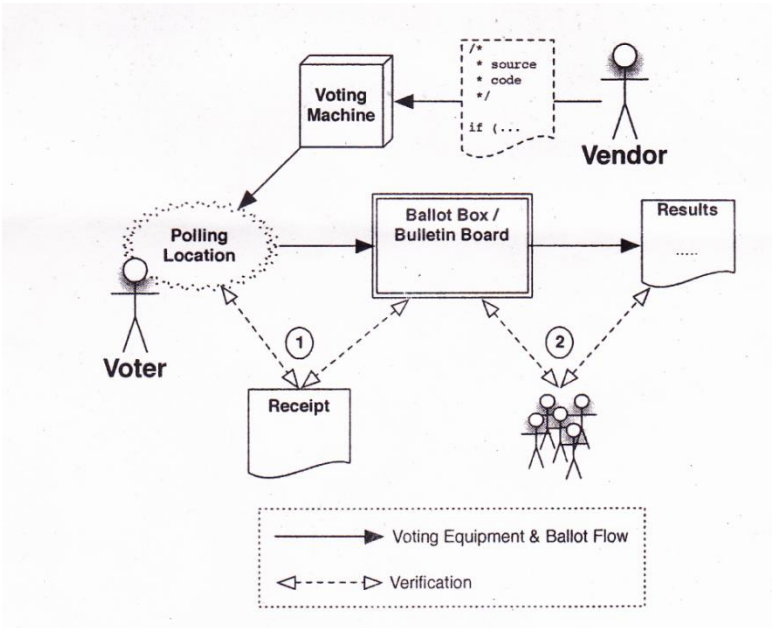


Figure **1-3: End-to-End Voting** - only two checkpoints are required. (1) The receipt obtained from a voter's interaction with the voting machine is compared against the bulletin board and checked by the voter for correctness. (2) Any observer checks that only eligible voters cast ballots and that all tallying actions displayed on the bulletin board are valid.

A. End-to-End Verifiability

When dealing with complex systems, software engineering has long relied on the "end-to-end" principle [152], where, in order to keep the system simple, the 'smarts" of the system are kept at higher levels of abstraction, rather than buried deep in the stack. For example, when routing packets on the Internet, very few assumptions are made about the underlying transport mechanism. Instead, checksums are performed by sender and recipient to ensure end-to-end integrity in the face of random mistakes, and digital signatures are app1ied to thwart malicious modifications of the data. No details of traffic routing need to be verified in either case; instead, a certain property is preserved from start to finish, regardless of what happens in between.

Though not all systems are amenable to such a design, *voting systems are.* Rather than completely auditing a voting machine's code and ensuring that the voting machine is truly running the code in question, end-to-end voting verification checks the voting machine's *output* only. Rather than maintain a strict chain-of-custody record of all ballot boxes, end- to-end voting checks tally correctness using mathematical proofs. Thus, the physical chain of custody is replaced by a mathematical proof of end-to-end behavior. Instead of verifying the *voting equipment,* end-to-end voting verifies the *voting results* [147].

As an immediate consequence, *one need not be privileged to verify the election.* In a chain-of-custody setting, one has to keep a close eye on the process to be certain of correct execution; only election officials can do this directly. In an end-to-end verifiable setting, anyone can check the inputs and outputs against the mathematical proofs. The details of the internal processes become irrelevant, and *verifiability becomes universal,* as diagrammed in Figure 1-3.

Cryptography makes end-to-end voting verification possible. At a high level, cryptographic voting systems effectively bring back the voting systems of yore. when all eligible citizens voted publicly, with tallying also carried out in public for all to see and audit. Cryptographic schemes augment this approach with:

1. *encryption to* provide ballot secrecy, and
2. *zero-knowledge proofs to* provide public auditing of the tallying process.

B. A Bulletin Board of Votes

Cryptographic voting protocols revolve around a central, digital *bulletin board.* As its name implies, the bulletin board is public and visible to all, via, for example, phone and web interfaces. All messages posted to the bulletin board are authenticated, and, it is assumed that any data written to the bulletin board cannot be erased or tampered with. In practice, implementing such a bulletin board is one of the more challenging engineering aspects of cryptographic voting, as one must worry about availability issues beyond data corruption, such as denial-of-service attacks for both data publication and access. There are, however. known solutions to this problem [110].

On this bulletin board, the names or identification numbers of voters are posted in plaintext, so that anyone can tell *who* has voted and reconcile this information against the public registry of eligible voters. Along with each voter's name, the voter's ballot is posted in encrypted form, so that no observer can tell *what* the voters chose. Two processes surround the bulletin board. The *ballot casting process* lets Alice prepare her encrypted vote and cast it to the bulletin board. The *tallying process* involves election officials performing various operations to aggregate the encrypted votes and produce a decrypted tally, with proofs of correctness of this process also posted to the bulletin board for all observers to see.

Effectively, the bulletin board is the verifiable transfer point from identified to de-identified ballots. Votes first appear on the bulletin board encrypted and attached to the voter's identity. After multiple transformations by election officials, the votes end up *on* the bulletin board, decrypted and now unlinked from the original voter identity. Whereas classic voting schemes perform a complete and blind hand-off-----i.e. the ballot is dropped into a box —, cryptographic voting performs a *controlled hand-off,* where individual voters can trace their vote's entry into the system, and any observer can verify the processing of these encrypted votes into an aggregate, decrypted tally. This process is illustrated in Figure 1-4

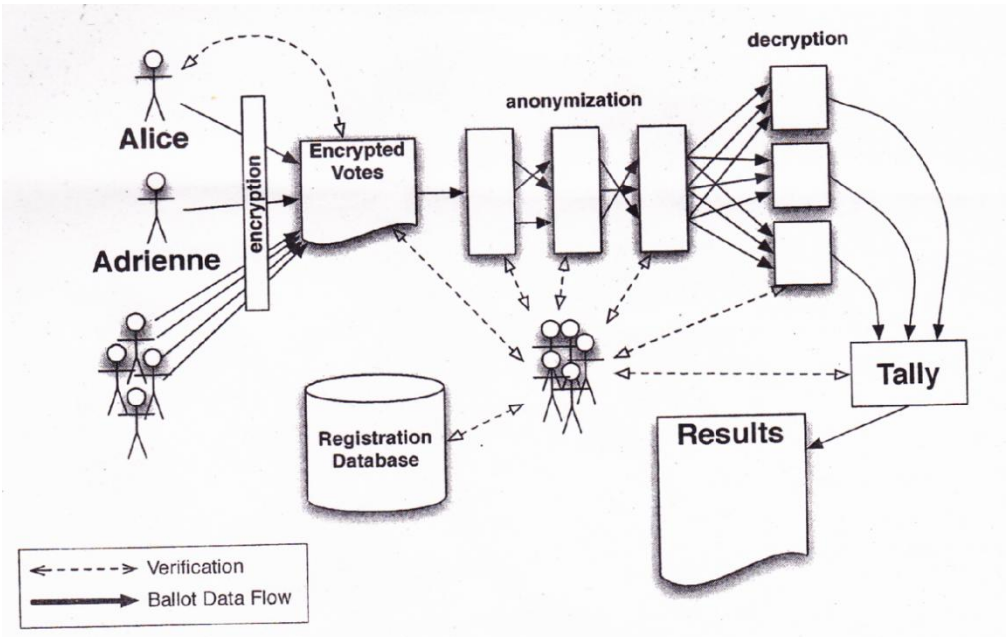


Figure **1-4: Cryptographic Voting at a High Level** - voters cast an encrypted ballot on a bulletin board, where voter names can be checked by anyone against a public voter registration database. Then, election officials proceed to anonymize the votes and jointly decrypt them, providing proofs along the way that any observer can verify. The results arc then posted for all to see.

C. A Secret Voter Receipt

Before Alice's encrypted ballot appears on the bulletin board, she must prepare it using some process that gives her personal assurance that her ballot has been correctly encrypted. Recall that, in providing Alice with this assurance, the system cannot enable her to transfer this same assurance to Carl, as this would enable Carl to influence her decision. For this purpose, all current cryptographic voting schemes require that voters physically appear at a private, controlled polling location: it is the only known way to establish a truly private interaction that prevents voter coercion.

In many proposed cryptographic schemes, Alice interacts privately with a computerized voting machine. She makes her selection much as she would using a classic voting machine, answering each question in turn, verifying her selections on screen, and eventually confirming her vote. The machine then produces an encryption of Alice's ballot, and begins a verification interaction with Alice.

This interaction is a type of *zero-knowledge proof,* where the machine proves to Alice that her encrypted vote indeed corresponds to her intended choice, without revealing exactly the details of this correspondence. In one particularly interesting scheme, Neff's MarkPledge.

[29], Alice obtains a printed receipt that includes her encrypted vote and some confirmation codes. By comparing the codes on the receipt with those on the screen of the machine, Alice can be certain that the machine properly encoded her vote. In addition, since Alice can easily claim, at a later point, that a different confirmation code appeared on the screen, she cannot be coerced. This scheme is diagrammed in Figure 1-5 and explored and extended in Chapter 5.

Using this encrypted receipt, Alice can verify that her encrypted ballot appears correctly on the bulletin board. Given that the ballot is encrypted, she can even give a copy of her receipt to helper political organizations — e.g. the ACLU, the NRA, the various political parties—so that they may verify, on her behalf, the presence of her encrypted vote on the bulletin board.

Paper-Based Cryptographic Voting. In 2004, Chaum [40] was the first to propose a cryptographic scheme that uses paper ballots. The ballot in question is specially formed: after filling out the ballot, Alice physically splits it into two pre-determined halves, destroys one, and casts the other while taking a copy of this same half home with her as a receipt. This separation effectively encrypts Alice's choice: oniy election officials with the proper secret keys can recover Alice's choice during the tallying process.

In most of these paper-based schemes, the paper ballot must be verified *prior to voting* to ensure that the two halves are consistent with one another. Without this verification, a fraudulently created ballot could corrupt the proper recording of the voter's intent. In Chaum's latest version, Punchscan [66], a second, post-election verification can also vetify the correctness of Alice's ballot. The details of the Chaum scheme are reviewed in Chapter 4, which also describes Scratch & Vote, our proposed evolution of Chaum's scheme, with methods to simplify the ballot face and pre-voting verification stage.

D. Tallying the Ballots

Once all encrypted votes are posted on .the bulletin board, it is time to tally them. Note that no single election official can decrypt these individual votes: the secret key required for decryption is shared among a number of election officials, who must collaborate for any decryption operation. This is extremely important, as any decryption at this point would violate the ballot secrecy of the voters in question. The decryption process must be well controlled to protect privacy.

Two major techniques exist for just this purpose. The first uses a special form of encryption-called homomorphic encryption—that enables the aggregation of votes under the covers of encryption, so that only the aggregate tally needs decryption. The second uses a digital version of "shaking time ballot box." where individual votes are shuffled and scrambled multiple times by multiple parties, dissociated from the voter identity along the way, and only then decrypted.

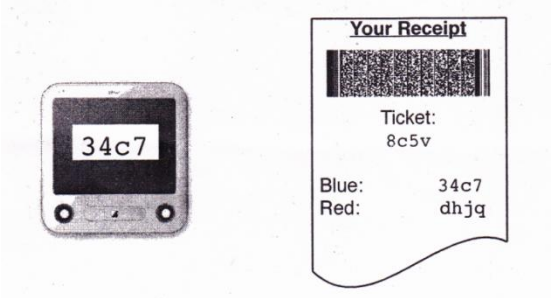


Figure 1-5: **A secret receipt in the Neff voter verification scheme** - The screen displays a code, which should match the voter's selected option on the receipt. In addition, the ticket number should match the voter's random challenge. Once the voter leaves the booth with only the receipt, it is impossible for her to provably claim that she saw 34c7' on the screen, and not dhjq.

Randomized Threshold Public-Key Encryption. Before we present either tallying method, it is important to note that all cryptographic voting systems use a special kind of encryption called *iandomized threshold public-key encryption.* The public-key property ensures that anyone can encrypt using a public key. The threshold-decryption property ensures that only a quorum of the trustees (more than the "threshold"), each with his own share of the secret key, can decrypt.

In addition, using *randomized* encryption, a single plaintext, e.g. Blue, can be encrypted in many possible ways, depending on the choice of a *randomization value* selected at encryption time. Without this property, since most elections offer only a handful of choices. e.g. Blue or Red, an attacker could simply try to deterministically encrypt all possible choices to discover, by simple ciphertext comparison, how everyone voted. With the randomization value, an attacker *would* have to try all possible random factors. Selecting a cryptosystem with a large enough set of randomization values ensures that this is never possible.

Tallying under the Covers of Encryption. Using a special form of randomized public- key encryption called homomorphic public-key encryption, it is possible to combine two encryptions into a third encryption of a value related to the original two, i.e. the sum. For example, using only the public key, it is possible to take an encryption of *x* and an encryption of *y* and obtain an encryption of $x + y$, all without ever learning *x* or *y or* $x + y$.

In a homomorphic voting scheme, as first proposed by Benaloh [43, 19], votes are encrypted as either 0 for Blue or 1 for Red. Then, all resulting encrypted votes are homomorphically added, one at a time, to yield a single encryption of the number of votes for Red. The trustees can then decrypt this single encrypted value to discover this tally for Red. The difference between the total number of votes and the Red tally is then the Blue tally. The approach can be extended to more than two options by encoding multiple "counters" within a single ciphertext, a technique we review in chapter 2. In addition, a zero-knowledge proof is typically required for each submitted vote, in order to ensure that each vote is truly the encryption of 0 or 1, and not, for example, 1000. Otherwise, a malicious voter could easily throw off the count by a large amount with a single ballot.

Homomorphic voting is particularly interesting because the entire homomorphic operation is publicly verifiable by any observer, who can simply re-compute it on his own using only the public key. The trustees need only decrypt a single encrypted tally for each election race. Unfortunately, homomorphic voting does not support write-in votes well: the encrypted homomorphic counters must be assigned to candidates before the election begins.

Shaking the Virtual Ballot Box. A different form of tallying is achievable using a *mixnet*, as first described by Chaum [39]. Mixnets typically use a *rerandomzable encryption scheme*, which allows anyone to take an encryption of a message and produce a new encryption of the same message with altered randomness. This is particularly interesting because, if one were simply to take a set of encryptions and reorder them, it would be trivial to compare the shuffled encryptions to the pre-shuffling encryptions. As randomized encryptions are effectively unique, the rerandomization property is necessary to perform true, indistinguishable shuffling.

In a mixnet. a sequence of mix servers, each one usually operated by a different political party, takes all encrypted votes on the bulletin board, shuffles and rerandomizes them according to an order and a set of randomization values kept secret. 'and posts the resulting set of ciphertexts back to the bulletin board. The next mix server then performs a similar operation, and so on until the last mix server. Then, all trustees cooperate to decrypt the individual resulting encryptions, which have, by now, been dissociated from their corresponding voter identity.

It is reasonable to trust that at least one of the mix servers will "shake the ballot box well enough" so that privacy is ensured. However, it is *not reasonable* to trust that no single mix server will replace an encrypted vote in the mix with a vote of its own. In other words, we trust the mix servers with the privacy of the vote, but we do not trust them with its correctness. Thus, each mix server must provide a zero-knowledge proof that it performed correct mixing, never removing, introducing, or changing the underlying votes. These types of proof are rather complicated, but a number of efficient schemes are known and implemented.

Mixnet-based voting is more difficult to operate that homomorphic-based voting, because the re-encryption and shuffle processes must be executed on a trusted computing base, keeping the details of the shuffle secret from all others. However. mixnets present two important advantages: the complete set of ballots is preserved for election auditing, and free-form ballots, including write-ins, are supported. As a result, mixnet-based voting schemes offer the most promise iii real-world. democratic election implementation. even if they are operationally more complex.

E. The Promise of Cryptographic Voting

With cryptographic voting promising significantly improved auditability, one might question why real elections have shunned these techniques to date. In fact, it is only recently that cryptographic voting schemes have become reasonably usable by average voters. Much necessary research remains to ensure that the extra voter verification processes can be made to work in a realistic setting. Most importantly, a significant education effort will be required, because the power of a cryptographically verified election is far from intuitive.

IV. Contributions of this Work

This paper contributes to the teaching, practice, and theory of cryptographic voting. Each contribution attempts to make cryptographic voting more useful and realistic.

A. Mixnet Literature Review

One critical component of a large number of cryptographic voting schemes is the anonymous channel, usually implemented by a robust, universally verifiable mixnet. The literature in this field spans 25 years without a single literature review. This dissertation (Chapter 3) presents just such a review of advances in mixnet research over the years, starting with Chaum's first mixnet in 1981 through the most recent mixnet work of early 2006.

B. Scratch & Vote

We propose Scratch & Vote, a paper-based cryptographic voting protocol. Inspired by the ballots of Chaum and Ryan [40, 41], S&V presents three significant practical advantages:

1.ballots and the ballot casting process use simple technology easily deployable today,

2. tallying is homomorphic, requiring only a single threshold decryption per race, and

3. ballots are self-certifying, which makes auditing and ballot casting assurance more practical.

Scratch & Vote uses 2D barcode to encode the encrypted votes and a scratch surface that hides the randomization values used to encrypt these ciphertexts. Thus, a ballot can be audited by simply removing the scratch surface, without contacting any central authority. Only ballots that have *not* been scratched off can actually be used in the vote casting: each voter takes two ballots, audits one and votes with the other. With a 50% chance of catching an error with each voter, any moderate attempt at fraud will be caught with just a handful of voter verifications.

C. Ballot Casting Assurance & Assisted-Human Interactive Proofs

We have seen that cryptography provides universal verifiability of the tabulation process. In addition, cryptography gives Alice two important capabilities: she can *verify directly* that her vote has been correctly cast, and she can *realistically complain* if she discovers an error. One contribution of this dissertation is the exact definition of these properties, and the coining of the term *ballot casting assurance* to describe them. In particular, ballot casting assurance supplants the unfortunately-abused term "cast as intended," which has been used with differing meaning by various voting research communities.

We then present *Assisted-Human Interactive Proofs (AHIP),* a definitional framework for interactive proofs where the verifier is significantly computationally limited -i.e. human. In the context of voting, this framework is particularly useful in describing how a voting machine proves to Alice that her vote was correctly encrypted. In particular, AHIP is the first definitional framework that captures the notion of a *secret voter receipt* and the security properties required of such a receipt.

We provide two implementations of an AHIP proof that a ciphertext encodes a specific option *j* out of *m* possible choices. The first protocol, a tweaked version of Neff's MarkPledge, provides ciphertexts whose length depends not only in the security parameter, but also in the desired soundness. The second protocol provides ciphertexts linear only in the security parameter. as long as soundness is within the range of the security parameter (it almost always is). Both schemes are particularly interesting because they are proven secure in this framework and demand very little from voters: they need only be able to compare very short strings--e.g. 4 characters.

D. Public Mixing

This paper also introduces *Public Mixing,* a new type of ballot-preserving anonymous channel. Typical mixnets await their inputs, then mix in private, then prove their mix. Public mixes prove their mix, then await their inputs, then mix through entirely public computation. In other words, all private operations and proofs in a public mix can be performed *before* the inputs are available. One should notice that this concept is by no means intuitively obvious. Public mixing creates, in a sense, the ability to obfuscate the program that performs the mixing. It is reasonable to think of public mixing as a combination of the advantages of homomorphic voting and mixnet voting: all election-day operations are public, yet all ballots are fully preserved.

We provide constructions of public mixing algorithms using the BGN [23] and Paillier [133] cryptosystems. The former case is simpler to explain but relies on a recently introduced cryptosystem with younger assumptions. The latter case is a significantly more complicated construction that relies on a much better understood set of assumptions and a cryptosystem that is well established in the literature.

Also covered in this work is the efficient distributed generation of a public mixer, because public mixing isn't compos able. In the voting setting, it is crucial that no single party know the anonymization permutation.

V. Universally Composable Mixnets

A mixnet is almost always a component of a larger, more complicated protocol, e.g. voting. As attacks over the years have shown, it is common to see a mixnet proven secure in a particular setting, only to find that it breaks down in a slightly different setting, e.g. if the mixnet is run twice with the same public key. The question; like for many other complex protocols, is how to precisely determine what the true security properties of a mixnet should be.

One approach to solving this problem generically is to prove security in the Universally Composable framework of Canetti [32], which we reviewed in Chapter 2. In this framework, a protocol proven secure can be composed with other secure protocols, even in parallel. In addition, the UC framework forces a high-level analysis of the security properties we desire from a mixnet.

A First Definition and Implementation

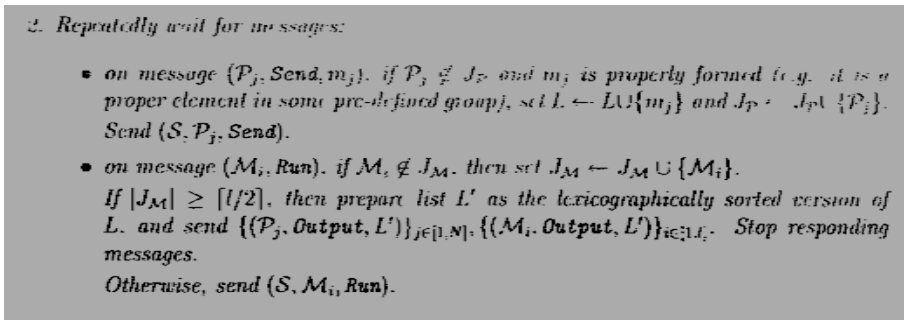
In 2004, Wikstrom [179] provided the first universally composable mixnet definition and implementation. The details of this new protocol are not particularly efficient, but the security proof is particularly interesting.

Defining the Ideal Functionality. Wikstrom gives F_{MN}, an ideal functionality for a mixnet. This functionality is particularly simple, relying on the. UC baseline to guarantee security: the senders submit their individual input to the ideal functionality. When a majority of the mix servers choose to "mix," the ideal functionality sorts the inputs lcxicographically and outputs the

resulting list. As is typical in the UC setting, the ideal adversary S can delay messages to the ideal functionality as it sees fit, though of course it cannot see the communication with the ideal functionality.

Definition 3-1 (Ideal Mixnet) *The ideal functionality for a mixnet, F_{MN}, with senders $P_{1,...}P_{,N}$ mix-servers M_1,.. . , M_2, and ideal adversary S, proceeds as follows: 1.Initialize:*

- *J_p = Ø the set of senders 'who have sent a message to the ideal functionality,*
- *J_m = Ø the set of mix servers who have sent a message asking to start the mixing*
- L = Ø a list of messages received from senders.



2. Repeatedly wait for messages:

- *on message $(\mathcal{P}_j, \textbf{Send}, m_j)$. if $\mathcal{P}_j \notin J_p$ and m_j is properly formed (e.g. it is a proper element in some pre-defined group), set $L \leftarrow L \cup \{m_j\}$ and $J_p \leftarrow J_p \cup \{\mathcal{P}_j\}$. Send $(\mathcal{S}, \mathcal{P}_j, \textbf{Send})$.*
- *on message $(\mathcal{M}_i, \textbf{Run})$. if $\mathcal{M}_i \notin J_{\mathcal{M}}$. then set $J_{\mathcal{M}} \leftarrow J_{\mathcal{M}} \cup \{\mathcal{M}_i\}$.*
  If $|J_{\mathcal{M}}| \geq \lceil l/2 \rceil$, then prepare list L' as the lexicographically sorted version of L. and send $\{(\mathcal{P}_j, \textbf{Output}, L')\}_{j\in[1..N]}, \{(\mathcal{M}_i, \textbf{Output}, L')\}_{i\in[1..l]}$. Stop responding messages.
  Otherwise, send $(\mathcal{S}, \mathcal{M}_i, \textbf{Run})$.

Wikström also gives UC definitions for an ideal bulletin board and an ideal distributed El Gamal key generation, suggesting implementations of the former by Lindell, Lysyanskaya and Rabin [110] and of the latter by Gennaro et al. [74]. We leave the details of these additional components to their respective publications and to Wikström's paper.

VI. Conclusion

The mixnet research field is rich with fascinating contributions to secure cryptosystems, efficient zero-knowledge proofs, and secure protocol definitions with the efficient proofs of Neff [124] and Furukawa and Sako [70], the efficiency of mixnets has become practical With the UC modeling work of Wikstrom, mixnets now have a solid formal model for solid security proofs. It remains to be seen, in the future, whether efficient implementations like Neff's can be adapted to frameworks like Universal Composability.

Mixnet Scheme	Based On	Attacks & Fixes	Fault-Tolerant	Verif.	Soundness	Privacy
Chaum Onions [39]	-	[138]	No	Sender	-	C&I
Park et al. partial dec. [134]	-	[136, 151]	No	Sender	-	C&I
Park et al. reenc. [134]	-	[136, 151]	No	Sender	-	C&I
Sako and Kilian [151]	[134]	[118], this review	No	Univ.	OW Proof	C&I
Ogata et al. [127]	[134, 151]	-	Yes	Univ.	OW Proof	C&I
Abe [1]	[134, 151]	-	Yes	Univ.	OW Proof	C&I
Jakobsson, Juels, Rivest [96]	[39]	-	-	Univ.	OW Proof †	C&I
Permutation Networks [2, 94]	[134]	-	Yes	Univ.	OW Proof	C&I
Furukawa & Sako [70]	[134]	-	Yes	Univ.	OW Arg	C&I
Neff [124]	[134]	-	Yes	Univ.	OW Proof	C&I
A Practical Mix [92]	[134]	[93, 55]	Yes	Quorum	-	C&I
Flash Mixing [93]	[134, 92]	[119, 178]	Yes	Quorum	-	C&I
Optimistic Mixing [84]	[134, 92]	[178]	Yes	Univ.	OW Arg	C&I
Almost Correct Mixing [26]	[134]	-	Yes	Univ.	HP Proof (α-dep.)	Incomplete
Parallel Mixing [83]	[134]	-	Yes	-*	-*	C & D
UC Mixnet [179]	[134, 55]	-	Yes	UC	OW Proof	C&I
Sender-Verifiable Mixnet [180]	[134, 179]	-	Yes	UC	OW Proof	C&I
Adaptively-Secure Mixnet [181]	[179]	-	Yes	UC	OW Proof	C&I

* These characteristics depend on the subprotocol chosen, which is independent of Parallel Mixing.

† Randomized Partial Checking has a slightly weaker definition of soundness, though one sufficient for voting.

Table 3.1: Summary of Mixnet Protocols. Each mixnet protocol is listed with the prior protocols from which it inherits, the papers that present flaws, the papers that present fixes, the verifiability and privacy properties. The protocols are ordered chronologically. The indicated privacy and soundness are indicated as per the *originally claimed* values, not the result of any potential breaks. We do not include the hybrid mixnets or universal reencryption in this comparison, as they are qualitatively different. C&I stands for "Complete and Indepenent," while C&D stands for "Complete and Dependent."

REFERENCES

[1] Masayuki Abe. Universally verifiable MIX with verification work independent of the number of MIX servers. In Nyberg [126], pages 437—447.

[2] Masayuki Abe. Mix-networks on permutation networks. In Kwok-Yan Lam, Eiji Okamoto, and Chaoping Xing, editors, ASIACRYPT, volume 1716 of Lecture Notes in Computer Science, pages 258—273. Springer, 1999.

[3] ACM. *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, 1990. ACM, 1990.*

[4] ACM. *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada.* ACM, 1994.

[5] ACM. PODC *2001, Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, August 26-29, 2001, Newport, Rhode Island,* USA. ACM, 2001.

[6] Ben Adida and C. Andrew Neff. Ballot Casting Assurance. In EVT *'06., Proceedings of the First Usenix/ACCURATE Electronic Voting Technology Workshop, August 1st 2006; Vancouver BC',Canada., 2000. Available online at http://www.usenix.org/ events/evt06/tech/.*

[7] Ben Adida and Douglas Wikström. How to Shuffle in Public Cryptology ePrint Archive, Report 2005/394, 2006. http://eprint.iacr.org/2005/394.

[8] Alan A. Reiter. Hong Kong residents asked to photograph voting for pro-Beijing candidates, August 2004. http://www.wirelessmoment.com/2004/08/hong_kong_resid.html.

[9] Andrew Gumbel. *Steal This Vote: Dirty Elections and the Rotten History of Democracy in America. Nation Books, July 2005.*

[10] Ari Shapiro. Absentee Ballots Go Missing in Florida's Broward County, October 2004. http://www.npr.org/templates/story/storyphp?storyId=4131522.

[11] Avi Rubin. An Election Day clouded by doubt, October 2004. http://avirubin. corn/vote/op-ed.html.

[12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Kilian [108], pages 1—18.

[13] Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical multi-candidate election system. In *PODC* [5], Pages 274—283.

[14] Donald Beaver. Foundations of secure interactive computing. In Feigenbaum [63], pages 377—391.

[15] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In Nyberg [126], pages 236—250.

[16] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Santis [154], pages 92—111.

[17] Ben Adida and C. Andrew Neff. Assisted Human Interactive Proofs: A Formal Treatment of Secret Voter Receipts, 2006. In preparation.

[18] Ben Adida and Ronald L. Rivest. Scratch & Vote: Self-Contained Paper-Based Cryptographic Voting. In Roger Dingledine and Ting Yu, editors, *ACM Workshop on Privacy in the Electonic Society. ACM, 2006. To Appear.*

[19] Josh Benaloh and Moti Yung. Distributing the power of government to enhance the power of voters. In *PODC*, pages 52—62. ACM, 1986.

[20] Josh Cohen Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In STOC [4], pages 544—553.

[21] Avrim Blum, Merrick Furst, Michael Kearns, and Richard J. Lipton. Cryptographic Primitives Based on Hard Learning Problems. In Stinson [165], pages 278—291.

[22] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103—112. ACM, 1988.

[23] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2.-DNF formulas on ciphertexts. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science, pages 325—342. Springer, 2005.*

[24] Dan Boneh, editor. Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, *Proceedings, volume 2729 of Lecture Notes in Computer Science. Springer, 2003.*

[25] Dan Boneh and Matthew K. Franklin. Efficient Generation of Shared RSA Keys (Extended Abstract). In Kaliski [103], pages 425—439.

[26] Dan Boneh and Philippe Golle. Almost entirely correct mixing with applications to voting. In Vijayalakshmi Atluri, editor, ACM Conference on Computer and Communications Security, pages 68—77. ACM, 2002.

[27] Stefan Brands. Untraceable off-line cash in wallets with observers (extended abstract). In Stinson [165], pages 302—318.

[28] C. Andrew Neff. Codebooks. Unpublished Manuscript.

[29] C. Andrew Neff. Practical High Certainty Intent Verification for Encrypted Votes. http://votehere.com/vhti/documentation/vsv—2.0.3638.pdf, last viewed on August 30th, 2006.

[30] C. Andrew Neff. Coerced Randomization, April 2006. Private conversations with and unpublished manuscript by Andy Neff.

[31] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In Kaliski [103], pages 455—469.

[32] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. *In .42nd Annual Symposium on Foundations of Computer Science, FOGS 2001, 14-17 October 2001, Las Vegas, Nevada, USA, Proceedings*, pages 136—145. IEEE Computer Society, 2001.. (Full version at Cryptology ePrint Archive, Report 2000/067, http://eprint.iacr.org, October, 2001).

[33] Ran Canetti and Rosario Gennaro. Incoercible multiparty computation (extended abstract*). In 37th Annual* IEEE Symposium on *Foundations of Computer Science* (FOCS 1996), 1996, Proceedings, pages 504—513. IEEE Computer Society, 1996.

[34] CBS News. How to Spend a Billion Dollars, May 2004. http://www.cbsnews. com/stories/2004/05/12/po1itics/main617059.shtml.

[35] CBS News. Switzerland Tries Internet Voting, September 2004. http://www. cbsnews.com/stories/2004/09/25/world/main645615.shtml.

[36] Charles Stewart III and Julie Brogan. Voting in Massachusetts, August 2003. http://www.vote.caltech.edu/media/documents/VotinginMass.pdf.

[37] D. Chaum and T. Pedersen. Wallet Databases with Observers. In Ernest F. Brickell, editor, CRYPTO, volume 740 of Lecture Notes in Computer Science, pages 89 105. Springer, 1993.

[38] David Chaum. Punchscan. http://punchscan.org viewed on August 13th, 2006.

[39] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM, 24(2):84—88, 1981.

[40] David Chaum. Secret-Ballot Receipts: True Voter-Verifiable Elections. IEEE Security and Privacy, 02(1):38—47, 2004.

[41] David Chaum, Peter Y. A. Ryan, and Steve Schneider. A practical voter-verifiable election scheme. In Sabrina Dc Capitani di Vimercati, Paul F. Syvcrson, and Dieter Gollmann, editors, ESORIGS, volume 3679 of Lecture Notes in Computer Science, pages 118—139. Springer, 2005.

[42] Richard Clayton. Improving onion notation. In Roger Dingledine, editor, Privacy Enhancing Technologies, volume 2760 of Lecture Notes in Computer Science, pages 81—87. Springer, 2003.

[43] Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme. In FOCS, pages 372—382. IEEE Computer Society, 1985.

[44] Ronald Cramer, Ivan Damgard, and Berry Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In Desmedt [52], pages 174—187.

[45] Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. Multi- authority Secret-Ballot Elections with Linear Work. In Ueli M. Maurer, editor, EUROCRYPT, volume 1070 of Lecture Notes in Computer Science, pages 72—83. Springer, 1996.

[46] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. Ill Walter Fumy, editor. EUROGRYPT, volume 1233 of Lecture Notes in Computer Science, pages 481—490. Springer, 1997.

[47] Ivan Damgard. Towards practical public key systems secure against chosen ciphertcxt attacks. In Feigenbaum [3], pages 445—456.

[48] Ivan Damgârd and Mats Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, Public Key Cryptography, volume 1992 of Lecture Notes in Computer Science, pages 119—136. Springer, 20001.

[49] Ivan Damgard and Maciej Koprowski. Practical Threshold RSA Signatures without a Trusted Dealer. In Pfitzmann [137], pages 152—165.

[50] David Jefferson. AvielD. Rubin. Barbara Simons. David Wagner. A Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE). http://www.servesecurityreport .org/.

