Coercion-Resistant Electronic Elections with Write-In Candidates

Carmen Kempka
kempka@kit.edu
Institute of Cryptography and Security, Faculty of Informatics,
Karlsruhe Institute of Technology, Germany

Abstract

It is often argued in the e-voting community that in the presence of write-in candidates, forced abstention attacks are always possible. Therefore, write-in candidates are often excluded in existing definitions of coercion-resistance arguing that those definitions cannot be achieved by write-in supporting schemes. This is only true if the tally is made public directly. Coercion-resistance may well be achieved if only a fuzzy version of the tally is published. This paper provides a formalization of fuzzy tally representations which enables definitions for coercion-resistance to take into account write-in candidates without being weakened. We also show how the cryptographic voting scheme Bingo Voting can be applied to write-in candidates with respect to this formalization, providing what we believe to be the first e-voting scheme that prevents forced abstention while allowing for write-in candidates. We then give a general construction of coercion-resistant schemes that provide a verifiable fuzzy tally representation from mix-based and homomorphic election schemes with trusted authority.

1 Introduction

Elections should be verifiable and coercion-resistant. Cryptography gives us the means to make it so, as the existence of many schemes shows, a few examples being [4, 9, 10, 20]. Several frameworks for electronic elections including several notions of coercion-resistance and voter-privacy exist [16, 24, 14, 12]. A remaining gap is that while in some elections, it is legally required that voters can vote for a write-in candidate, existing definitions of coercion-resistance disregard or even exclude write-in candidates, arguing that as soon as write-in candidates come into play, forced abstention attacks are always possible by forcing the voter to write in a certain rarely elected candidate or a certain random string, therefore an election allowing write-in candidates can never be fully coercion-resistant. As stated by Juels et. al.[16] this is true if the tally is published directly. But it is actually possible for write-in supporting schemes to be fully coercion-resistant if only a fuzzy version of the tally is published. In real world elections a fuzzy representation of the tally is often sufficient. Examples for such fuzzy representations are the representation of the tally in percent or the resulting number of parliamentary seats for each candidate. In this case, if a candidate only got zero or one votes, it is sufficient that the representation of the tally shows that the number of votes for this candidate is less than a certain threshold, instead of showing whether this candidate got a vote or not. However, the added fuzziness should not be greater than necessary. This work takes a closer look at this fuzziness.

Contributions: We provide a formalization of fuzzy tally representations that enables reasonable definitions of coercion-resistance to be achieved by write-in supporting verifiable election schemes without weakening that definition. One problem is that if the tally is published fuzzy, a verifiable election scheme has to prove the correctness of the fuzzy tally without revealing the exact tally. We provide the cryptographic voting scheme Bingo Voting with the possibility to allow for write-in candidates. Thereby we provide what we believe to be the first cryptographic voting scheme with verifiable correctness that allows for write-in candidates while achieving a feasible notion of coercion-resistance and provably preventing forced abstention. We then provide general constructions of verifiable voting schemes supporting fuzzy tally representations from verifiable mix-based and homomorphic schemes with a trusted voting authority.

This work can be seen as an add-on to existing and upcoming models, bridging the gap between write-in candidates and coercion-resistance. The idea of this work is as follows: Consider a voting scheme with verifiable correctness that is coercion-resistant (by any definition), and provides a proof of correctness of the tally which can be adapted such that it proves the correct-

ness of the fuzzy tally representation without revealing the exact tally. Applied to an election where the tally can be published fuzzy according to our definitions of fuzzability, the adapted scheme should still achieve the same coercion-resistance while additionally being resistant against forced abstention caused by forcing voters to vote for a rarely elected candidate. This holds even if the scheme supports write-in candidates, assumed that a voter can add names to the representation of the tally without voting for (or losing her vote for) them.

Some remarks: On a first glance, there seems to be a simple countermeasure to the attack that a voter is forced to vote for an unlikely or fictional write-in candidate in opening and counting write-in votes only if they could make a difference in the result. On a closer look, this is not as trivial as it seems. To prevent voters from being coerced to write-in a name instead of voting for a list candidate, the voting process must not leak the information if a voter voted for a write-in candidate or not. At the same time, to preserve verifiable correctness, there needs to be a proof that those unopened votes are really for write-in candidates and that they really make no difference in the result. This is especially a problem if voters can validly write in names of list candidates. By contrast, the modification of Bingo Voting presented here proves the correctness of the tally without revealing any information about a single voter's choices.

There are some attacks that are hard to prevent by any voting scheme and not considered in this paper. Forced abstention attacks are trivially possible in non-remote elections by observing the polling stations during the whole voting phase, and as Benaloh stated in [4] it is infeasible to prove that no camera is in the voting booth. We argue that these kind of attacks are technically much more complex than the evaluation of data on a website and are ineffective for large-scale attacks. Pattern-voting attacks are also not considered here, but as mentioned in [23], those can usually be prevented by contest splitting.

This paper is organized as follows: After describing related work in Section 2, we will formalize fuzziness in Section 3. Section 4 adapts the cryptographic voting scheme Bingo Voting to our notion of fuzziness. In section 5 Bingo Voting is provided with the possibility to include write-in candidates. Section 6 describes general constructions for verifiable mix-based and homomorphic schemes with fuzzy tally representation. Section 7 concludes this work and states some open problems and future work.

2 Related Work

Gardner et. al. [14] propose a game based definition of coercion-resistance based on the indistinguishability of protocol behavior given inputs of the voter's choice and inputs of the adversary's choice. They explicitly don't take into account write-in votes. Küsters et. al [18] provide a very abstract game-based definition of coercion-resistance. Like the definition of Gardner et. al. this definition is based on the indistinguishability of the two cases that 1) the voter behaves as told by the adversary and 2) the voter deceives the adversary with some counter strategy and casts a vote of her own choice. Using this definition, they prove that Bingo Voting is as coercion-resistant as an ideal voting scheme except for forced abstention. Juels et. al. [16] propose a model for electronic elections including definitions of verifiability and coercion-resistance. The authors state that forced abstention by forcing to vote for an unlikely candidate cannot be prevented as long as the tally is made public directly, and refrain from considering this attack in their model. In their work about ranking votes [24], Teague et. al. propose a definition for coercion-resistance that takes into account the information leakage of (intermediate) tally results.

Several schemes exist [1, 17] that efficiently include write-in votes in elections based on homomorphic encryption. While forced-abstention attacks are not considered in [17], Acquisti [1] prevents forced abstention attacks based on forcing the voter to "vote for" a random string by only allowing certain permissible choices. This does not keep an adversary from forcing the voter to vote for a valid, rarely elected candidate. As stated in [2], mixnet-based schemes like Neff's [21] naturally support write-in votes as they support free-form ballots. Scantegrity II [9] has been used in a real election [8] providing the possibility to write in candidates in a way which is practical and straightforward for the voter. Their technique is similar to ours in that it also tallies in two steps: first the candidates from the list and then the write-in candidates. However, the write-in solution of Scantegrity II has some drawbacks. As stated in [8], the election authority can modify the write-in names. Furthermore, any observer auditing the resolution of write-in votes sees the handwritten choices. The two-step idea of counting write-in votes separately has also been proposed in [2] for paper-based schemes. All schemes allowing write-in votes, including those mentioned above, are naturally vulnerable to forced-abstention attacks as long as their proofs of correctness yield the exact tally.

Bohli et. al. [6] propose additional features for Bingo Voting (and other voting machines). One of them is linking the receipts with a chain of hash values to protect their integrity, this does not change the basic scheme and can also be done in our proposed scheme. They also propose proving correctness of intermediate results directly after each voting process and then immediately deleting all secret information needed for this proof, maximizing voter secrecy if the voting machine is corrupted during

the voting phase. Their approach is not directly applicable to this work, which therefore refers to the original scheme as described in [7]. To the author's knowledge, no technique for Bingo Voting to support write-in votes has been proposed yet.

3 Formalizing Fuzziness

In the literature it is usually assumed that everyone including the adversary sees the whole tally, and therefore, abstention attacks are always trivially possible by forcing a voter to vote for an unlikely candidate. This is why existing definitions of coercion-resistance do not consider this attack. For the same reason, as soon as write-in candidates are supported, coercion-resistance as defined in the literature seems impossible to achieve.

As a solution to prevent forced abstention attacks in the presence of write-in candidates (or very rarely elected candidates), we propose publishing the tally in a fuzzy way that does not weaken verifiability more than necessary. Hence published proofs of correctness will not prove the exact tally, only that the tally is very close to the published one. This section first provides a rather strong definition of fuzziness and then a weaker version of this definition in which the fuzziness/accuracy-tradeoff is adjustable in a more fine-grained way.

Since the author sees this formalization is an add-on to existing (or upcoming) models, this work refrains from giving a specific definition for an election system. Neither do we commit ourselves to a certain definition of coercion-resistance or a corruption model.

3.1 Defining Fuzziness

We differentiate between the *tally* and its *representation*. We assume the tally itself to remain secret, while instead a representation of the tally is published.

We denote the set of possible tally outcomes of a given election by $\mathscr{T}$. A *representation R* can be seen as a view that represents a certain subset $T_R \subset \mathscr{T}$ of tallies.

Please note that possible representations of the tally are co-determined by the election since the election defines the needed accuracy with which the tally has to be represented.

> **Definition 1 (complete)** *We call a set $\mathscr{R}$ of representations* complete *for a set of tallies $\mathscr{T}$ if $\bigcup_{R\in\mathscr{R}} T_R = \mathscr{T}$ where T_R denotes the set of tallies represented by a representation $R \in \mathscr{R}$.*

Hence a set of representations is complete if each possible tally has a representation.

We henceforth assume that the tally is a vector of length n where n is the number of candidates. The ith entry in this vector corresponds to the number of votes for the ith candidate.

> **Definition 2 (δ-neighbored)** *Let A and B be two vectors of length n. We call A δ-*neighbored *to B, if a vector $X = (x_1, x_2, \ldots, x_n)$ exists with $\sum_{i=1}^{n} |x_i| \leq \delta$ and $A = B + X$.*

> **Definition 3 (μ, δ-fuzzability)** *Let $\mathscr{T}$ be the set of all possible tallies of a given election. We call this election μ, δ-*fuzzable *if there is a set $\mathscr{R}$ of representations that is complete for $\mathscr{T}$ and for each representation $R \in \mathscr{R}$ the following holds for its set of represented tallies $T_R \subseteq \mathscr{T}$:*
>
> 1. *For each pair of tallies $(T_1, T_2) \in T_R \times T_R$ it holds that T_1 is δ-neighbored to T_2.*
> 2. *Let E_i be the set of all entries that occur at position i within any tally-vector in T_R. Then $|E_i| \geq \mu$ for $i = 1, \ldots, n$.*

Intuitively, the second requirement means that there are at least μ different tally outcomes for each candidate encoded in each representation.

> **Definition 4 (μ, δ-fuzzy election scheme)** *We call an election scheme μ, δ-*fuzzy*, if applied to a μ, δ-fuzzable election with a set of possible tallies $\mathscr{T}$, a corresponding set $\mathscr{R}$ of representations with corresponding represented subsets T_R of $\mathscr{T}$ according to the definition of μ, δ-fuzzability, for each representation $R \in \mathscr{R}$ representing tally $T \in T_R$ its proof of correctness proves that $T \in T_R$ and no other information than just that is revealed by any public data produced in the election process.*

Particularly, if $\mu > 0$ the proof of correctness must not yield the tally itself. This means that voting schemes whose proofs of correctness can only be verified with knowledge of the exact tally cannot comply with the above definition unadapted.

3.2 Weak Fuzziness

To prevent forced abstention, full μ, δ-fuzzability is not generally required (though it makes the construction of a corresponding voting scheme easier). We usually do not have to conceal if a candidate got, say 150 or 151 votes. In some cases we even must not conceal this because the order of the most elected candidates is an important tally outcome. If we just want to conceal if a candidate got zero votes or more, a weaker definition as follows is sufficient.

Definition 5 (weak μ,δ-fuzzability) *Let $\mathscr{T}$ be the set of all possible tallies for a given election. We call this election* weak μ,δ-fuzzable*, if there is a set $\mathscr{R}$ of representations that is complete for $\mathscr{T}$ and the following holds for each tally $T \in \mathscr{T}$: Let $T = (x_1,\ldots,x_n)$. For each representation $R \in \mathscr{R}$ that represents T, meaning $T \in T_R$, the following holds:*

1. *For each pair of tallies $(T_1,T_2) \in T_R \times T_R$ it holds that T_1 is δ-neighbored to T_2.*
2. *For each $i \in \{1,\ldots,n\}$ with $x_i < \mu$: Let E_i be the set of all entries that occur at position i within any tally-vector in T_R. Then $|E_i| \geq \mu$.*

Intuitively this means that if a candidate got less than μ votes, then each representation of the tally indicates μ other possible numbers of votes for this candidate.

Please note that μ,δ-fuzzability implies weak μ,δ-fuzzability: Given μ,δ-fuzzability, for each representation R there occur μ different outcomes in T_R for each candidate, in particular for those having less than μ votes.

Definition 6 (weak μ,δ-fuzzy election scheme) *We call an election scheme* weak μ,δ-fuzzy*, if applied to a weak μ,δ-fuzzable election with a set of possible tallies $\mathscr{T}$ and a corresponding set $\mathscr{R}$ of representations according to the definition of weak μ,δ-fuzzability, for each representation $R \in \mathscr{R}$ representing tally $T \in T_R$ its proof of correctness proves that $T \in T_R$ and no other information than just that is revealed by any public data produced in the election process.*

For real world elections, parameters δ and μ less than 5 seem reasonable.

4 Fuzzy Bingo Voting

As an example, we adapt Bingo Voting to our notion of fuzziness and show that the adapted scheme is resistant against forced abstention by forcing a voter to vote for an unlikely candidate. In Section 4.1 we briefly describe the original Bingo Voting scheme. In Section 4.2 we describe a version of Bingo Voting that is μ,μ-fuzzy for arbitrary μ. To allow for a more precise tally representation, we propose a weak μ,μ-fuzzy version of Bingo Voting in Section 4.3, where the tally can be published directly if there is no candidate who got less then μ votes. In this scheme, if a candidate got less then μ votes, only the number of votes of this candidate and one other has to be made fuzzy.

4.1 The Original Bingo Voting Scheme

In this section we briefly describe the original Bingo Voting scheme. For a more detailed description see for example [7].

Bingo Voting is a cryptographic computer-assisted voting scheme which relies on a trusted random number generator. It provides verifiable correctness, and if the voting computer is uncorrupted it also provides secrecy.

The scheme is rather flexible since it allows cross voting and cumulation of votes, but it does not support write-in candidates as it is. We will fix this in Section 5.1. The scheme consists of three phases that will be described in more detail in the following sections. The pre-voting phase consists of the precalculation and publishing of commitments to so-called dummy votes in the form of random numbers that are later used on the voter's receipt to conceal the voter's choice. After entering her vote, the voter gets a receipt that contains dummy votes for each not elected candidate and for each elected candidate a fresh random number from the trusted random number generator which represents the voter's choice but is indistinguishable from the dummy votes. In the post-voting phase, the receipts, the tally and its proofs of correctness are published for verification.

4.1.1 Preconditions

All calculations including the creation of dummy votes and all proofs of correctness are done by the voting computer. Preconditions for the correctness of the tally are the fact that the random number generator is untampered with and has its own (also uncorrupted) display, as well as the existence of a public bulletin board which can be thought of as a website, meaning anyone has read access and an election authority that doesn't have to be trusted for correctness has write access. For secrecy we additionally have to assume that the voting computer is uncorrupted.

We model this as an election authority that has full access to the voting computer and write access to the bulletin board. This election authority then has to be trusted for secrecy but not for correctness.

4.1.2 Pre-Voting Phase

In the pre-voting phase, dummy votes for each candidate are created that are later used on the voter's receipt to conceal her choice. Each candidate gets the same number k of dummy votes, where k has to be (at least) the number of votes that can be cast for one candidate during the voting phase. Let n be the number of candidates, $\mathscr{C} = \{C_1,\ldots,C_n\}$ the set of candidates and l the number of eligible voters. For the sake of perspicuity let us

assume that each voter can cast one vote for one candidate, so we have a one out of n choice. This means that at least $k = l$ dummy votes for each candidate $C_1, \ldots, C_n$ have to be created. More dummy votes can be created if the exact number of expected voters is unclear, as long as this is later accounted for in the calculation of the tally. For now we assume that exactly $k = l$ dummy votes are created.

The dummy votes are created as follows: For each candidate $C_i \in \mathscr{C}$, k random numbers $r_{C_i,1}, \ldots, r_{C_i,k}$ are drawn, these random numbers are the dummy votes for candidate C_i. Those random numbers have to be mutually distinct and indistinguishable from those drawn by the trusted random number generator during the voting phase, so they are ideally drawn out of the same source.

For each candidate C_i, commitments $com(C_i, r_{C_i,j})$ to each of his dummy votes $r_{C_i,j}$ are created. The commitments are published, the random numbers themselves must be kept secret by the voting authority. Additionally a proof is published that each candidate has the same number of dummy votes. In the original scheme [7] those proofs are done with randomized partial checking [15], but the authors of [7] also state that other proof techniques can be used. Details of this proof or the exact proof technique are not relevant for our work as long as the proof is sound and done without revealing the random numbers, so we refrain from presenting a proof here. As commitments, the original Bingo Voting scheme uses Pedersen commitments [22], because they are unconditionally hiding and maskable through rerandomization. They can also be used in the adapted schemes introduced in this work, if the preelection proof is also done with randomized partial checking.

4.1.3 Voting Phase

In the voting booth, the voter uses the voting computer to enter her choice. She then gets as a printed receipt a list of all candidates with a random number behind each candidate's name. Each candidate except the elected one has a dummy vote written behind his name, this dummy vote is deleted from the candidate's pool of dummy votes and not used again on another receipt. The elected candidate should instead have a fresh random number from the trusted random number generator written behind his name. In the privacy of the voting booth, the voter can check the equality of the number behind the elected candidate's name and the fresh random number shown on the trusted random number generator's display. She can take the receipt with her and forget the fresh random number after checking. When the voter leaves the voting booth, the random number generator's display is cleared, and the receipt does not show which of the numbers were fresh, this information is only known to the voter and the voting authority and cannot be proven by the voter to an adversary.

4.1.4 Post-Voting Phase

After the election is closed, the tally and all receipts are published on the public bulletin board, and the voters can check the presence of their receipts. All commitments to unused dummy votes are opened publicly. The unused dummy votes mirror the tally, since each time a candidate gets a vote, he does not lose a dummy vote. It is thus sufficient to publish the unused dummy votes as the tally. What is also published is a proof that each receipt contains a) exactly one fresh random number and b) for each not elected candidate a dummy vote of which a commitment was published in the pre-voting phase. The proof takes as input a new commitment to the fresh random number and the commitments to the dummy votes on the receipt which were published in the pre-voting phase. It is then proven that there is one commitment for each candidate and that the content of the commitments correspond to the random numbers on the receipt, without revealing any associations between commitments and random numbers. The details of this proof are again omitted here since there are different proof techniques [7, 15] that work here as well as with our adapted fuzzy scheme which is described in Section 4.2.

4.2 The Fuzzy Scheme

We define the tally T as well as its representation R to be vectors of length n where n is the number of candidates. The ith entry in T corresponds to the number of votes for the ith candidate and the ith entry in R is the representation of the number of votes for the ith candidate.

In the following we describe a Bingo Voting scheme that is μ, μ-fuzzy for arbitrary parameters μ. For the sake of perspicuity we first describe a scheme not supporting write-in candidates. We will describe a write-in supporting μ, μ-fuzzy version of Bingo Voting in Section 5.2.

The idea is as follows: In the pre-voting phase, we prepare enough dummy votes that in any case at least μ dummy votes of each candidate will remain unused after the election. In the post-voting phase, we do not open all unused dummy votes, just almost all. The opened unused dummy votes are our representation of the tally. The set of remaining unopened dummy votes is filled up with some new commitments. Of this filled-up set of commitments we prove that there are at most μ commitments for each candidate, so each candidate C_i has at most μ more votes than in the ith entry in the representation (minus μ, since we created μ more dummy votes than needed). In particular, in this scheme it is proven that a rarely elected

candidate has less than μ votes without revealing if he got a vote or not.

Consider a μ,μ-fuzzable election with l eligible voters and one vote per voter, n candidates $C_1,\ldots,C_n$, and a set

$$\mathscr{R} = \{(R_1,\ldots,R_n) | 0 \leq R_i \leq l+\mu \text{ for } 1 \leq i \leq n\}$$

of possible representations. With one vote per voter, l is the maximum number of votes a candidate can get during the election. The set T_R of tallies represented by an $R \in \mathscr{R}$ with $R = (R_1,\ldots,R_n)$ is

$$T_R = \{(T_1,\ldots,T_n) | R_i - \mu \leq T_i \leq R_i \text{ for } 1 \leq i \leq n\}.$$

4.2.1 Pre-Voting Phase

The pre-voting phase is done as described in Section 4.1, except that for each candidate at least μ more dummy votes than needed are prepared. So in an election with n candidates $C_1,..,C_n$ and l eligible voters, at least $k = l+\mu$ dummy votes $r_{C_i,1},\ldots,r_{C_i,k}$ for each candidate C_i are prepared. As in the original scheme, commitments $com(C_i, r_{C_i,j}), i = 1,\ldots,n, j = 1,\ldots,k$ to these dummy votes are published and it is proven that each candidate has the same amount of dummy votes.

4.2.2 Voting Phase

The voting phase is exactly the same as in the original scheme, described in Section 4.1.3.

4.2.3 Post-Voting Phase

We make use of the fact that for secrecy we need to assume a trustworthy voting authority, so we have an instance who can open commitments and compute the tally in secret without making additional assumptions. Please note that if secrecy does not hold, forced-abstention and other coercion attacks are trivially possible.

Remember that the tally is mirrored by the unused dummy votes, this time with an offset of μ because μ more dummy votes for each candidate were created in the pre-voting phase. Let $T = (T_1,..,T_n)$ be the tally, then there are $T_i + \mu$ unused dummy votes for each candidate C_i. Again, all receipts are published with corresponding proofs that the number of fresh random numbers is correct, this part is the same as in the original scheme. But this time, not all unused dummy votes are opened. Instead, a vector $f = (f_1,\ldots,f_n)$ is chosen by the voting authority at random but with the following restrictions:

1. $\sum_{i=1}^{n} f_i \geq \mu$
2. $0 \leq f_i \leq \min\{T_i, \mu\}$ for all i

The vector f determines in which way the tally is made fuzzy, and must be kept secret. As a representation of the tally, we publish the vector $R = (R_1,\ldots,R_n)$ where

$$R_i = T_i + \mu - f_i \text{ for } i = 1,\ldots,n.$$

For each candidate C_i, the commitments to R_i of his unused dummy votes are opened, so f_i commitments to C_i's dummy votes remain unopened. This set of unopened commitments is now filled up in the following way: For each candidate C_i, $\mu - f_i$ new commitments of the form $com(C_i, r_j), j = 1,\ldots,\mu - f_i$ are created, where the r_j are new random numbers that are mutually distinct and indistinguishable from the dummy votes. These new commitments together with the unopened commitments to dummy votes are published on the public bulletin board, so everyone can see how many new commitments were created, but not how many for which candidate. These commitments are then opened in a way that hides the association between commitment and content but proves that the set of published contents corresponds to the set of published commitments. This is the same kind of proof that is used in the proof that each receipt contains one fresh random number and can be done with the same proof technique, for example randomized partial checking. Everyone can check that of those opened commitments, there are μ for each candidate. This proves that each candidate has got at most μ votes more than indicated by the representation.

To prevent forced abstention, it is sufficient to just make entries of the tally vector fuzzy that are smaller than μ. We therefore provide a weak μ,μ fuzzy scheme in Section 4.3.

4.2.4 Fuzziness of the Scheme

We now discuss why the scheme presented above is μ,μ-fuzzy. In the post-voting phase, a representation $R = (R_1,\ldots,R_n)$ of the tally $T = (T_1,\ldots,T_n)$ is published, and a proof is given that the tally lies between $(R_1-\mu,\ldots,R_n-\mu)$ and $(R_1,\ldots,R_n)$. No more information about the tally than this is yielded by any published data. So the set T_R of tallies represented by R consists of all vectors $(x_1,\ldots,x_n)$ where $R_i \leq x_i \leq R_i + \mu$ for all $i = 1,\ldots,n$. This set indicates μ different possible tally outcomes for each candidate and obviously all vectors in T_R are μ-neighbored.

4.3 Bingo Voting with Weak Fuzziness

Since it is not always convenient and necessary to make the whole tally fuzzy, we propose a weak μ,μ-fuzzy scheme applied to a (not weak) μ,μ-fuzzable election. In this scheme, the accuracy of the representation of tally

can be adjusted more precisely. Like in the previous section, consider a μ,μ-fuzzable election with l eligible voters and one vote per voter, n candidates $C_1,\ldots,C_n$, and a set

$$\mathscr{R} = \{(R_1,\ldots,R_n) | 0 \leq R_i \leq l+\mu \text{ for } 1 \leq i \leq n\}$$

of possible representations. Each $R = (R_1,\ldots,R_n) \in \mathscr{R}$ represents the tallies in the set

$$T_R = \{(T_1,\ldots,T_n) | R_i \leq T_i \leq R_i+\mu \text{ for } 1 \leq i \leq n\}.$$

The pre-voting phase and voting phase are the same as in the μ,μ-fuzzy scheme described in the previous section. In the post-voting phase, the trusted authority calculates the tally $T = (T_1,\ldots,T_n)$ in secret. Similar to the μ,μ-fuzzy scheme the voting authority chooses a vector $f = (f_1,\ldots,f_n)$ such that $0 \leq f_i \leq \min\{\mu, T_i\}$ for $i = 1,\ldots,n$ and $\sum_{i=1}^{n} f_i \geq \mu$. The difference to the previous scheme is that now a second vector $d = (d_1,\ldots,d_n)$ is chosen such that

1. $f_i + d_i \leq \mu$ for all i
2. $f_i + d_i = \mu$ for each i where $T_i \leq \mu$,
3. if an i exists with $T_i \leq \mu$, then
   (a) $\sum_{i=1}^{n} d_i \geq \mu$
   (b) $f + d$ must have at least two entries that are greater than zero.

The vectors f and d must be kept secret. As in the μ,μ-fuzzy scheme, the representation $R = (R_1,\ldots,R_n)$ with $R_i = T_i + \mu - f_i$ for all i is published. We open R_i of the unopened commitments for each candidate C_i, which proves that C_i got at least $R_i - \mu$ votes. Now f_i commitments remain unopened for each candidate C_i. The voting machine creates d_i new commitments $com(C_i, r_{C_i,j}), j = 1,\ldots,d_i$ for each candidate such that all $r_{C_i,j}$ are mutually distinct and indistinguishable from the dummy votes. We now have $f_i + d_i$ unopened commitments for each candidate C_i. Those commitments are published and opened in a way that hides the association between commitment and content but proves that the set of published contents corresponds to the set of published commitments. Again this can be done for example with randomized partial checking [15]. Please note that while f and d must remain secret, $f + d$ becomes public. So the published data indicates that each candidate C_i got between $R_i - \mu$ and $R_i - \mu + f_i + d_i$ votes. This means that the number of votes for candidates C_i for which $f_i + d_i = 0$ is represented accurately, it is exactly $R_i - \mu$, each voter can check this using the data published on the bulletin board. Since $f_j + d_j = \mu$ for candidates C_j who got less then μ votes, the published data proves that C_j got between $R_j - \mu$ and R_j votes without revealing more information about this candidate's number of votes than just that. For all candidates C_h that got more than μ votes, $f_h + d_h$ lies between 0 and μ, so this candidate's number of votes may or may not be represented fuzzy, since $f_h + d_h$ is public the degree of fuzziness is public, too.

It is important that $f + d$ has to have either zero or at least two entries $\neq 0$. The reason for this is that if exactly one candidate C_x got $v \leq \mu$ votes and $f_x + d_x = \mu$ but $f_i + d_i = 0$ for all $i = 1,\ldots,n, i \neq x$, in the end only unopened dummy votes of candidate C_x remain. These are then filled up with other commitments for C_x and opened. So the published data indicates that all unopened commitments created and published in the pre-voting phase are for candidate C_x, and everyone can see how many there were, so the exact number of votes for C_x can be calculated. Besides, if the overall number V of votes cast in the election is published and we know that only the number of votes of C_x is made fuzzy, the number of votes for C_x can easily be calculated by subtracting all $T_i, i \neq x$ from V.

The advantage of our weak μ,μ-fuzzy scheme is that if a candidate got less than μ votes, only the number of votes for this candidate and one other has to be made fuzzy.

5 Bingo Voting with Write-In Support

In this section, we provide Bingo Voting with write-in support. For the sake of perspicuity we first describe a scheme without fuzziness in Section 5.1. Finally in Section 5.2 we propose a μ,μ-fuzzy version of the scheme with write-in support, providing what we believe to be the first verifiable election scheme with write-in support that is provably resistant against forced abstention caused by forcing voters to vote for an unlikely or fictional candidate.

5.1 Without Fuzziness

In this chapter, we show how Bingo Voting can be adapted to allow write-in candidates by the means of known techniques. This is done with almost no additional work for the voter. In fact, a voter who does not write in a candidate can do exactly the same as in the original scheme. Apart from a voluntary testing of the voting process her only additional effort compared to a pen-and-paper election is the also voluntary comparison of a random number. In addition our write-in technique is generic. We strongly suppose that write-in votes can be included in other electronic voting schemes in basically the same way.

The idea is to have an additional candidate named "Write-In" which is treated as a regular candidate. The "normal" tally, along with the votes for the list candidates, then yields the overall number of write-in votes. The actual votes for the write-in candidates are counted in a separate tally.

5.1.1 Preconditions

The same preconditions hold as in the original scheme described in Section 4.1. Since we want to prevent forced abstention we assume a trusted authority with access to an uncorrupted voting computer.

5.1.2 Pre-voting Phase

To be able to include a write-in candidate into the Bingo Voting election process, we use an additional mock candidate named "Write-In" as a wildcard character. Before the election starts, "Write-In" is added to the candidate list. Then the pre-voting phase is performed as described in Section 4.1. "Write-In" is treated as a regular candidate and therefore gets the same number of dummy votes as any other candidate. For the sake of simplicity we describe our technique for one write-in candidate. If the voter can write in more than one candidate, our scheme can be extended in a straightforward way by adding more mock candidates to the list.

5.1.3 Voting Phase

In the polling booth, at first nothing changes for the voter compared to the original scheme except that she now has the possibility to write in a candidate. The receipt, sketched in Figure 1, looks like a receipt of the original scheme, except that it now additionally contains a commitment to the write-in candidate's name. If the corresponding vote is not for a write-in candidate, the commitment is to the string "No Write-In".

After the voter has made her choice, either in the conventional way or by writing in a candidate, the printer begins to print the voter's receipt. First, only the commitment is printed. Now the voter has to be assured that this commitment contains the right name. To prevent coercion, the voter cannot just be given the unveil information. Instead, we use voter-initiated auditing [5]: The voter can choose to either test the voting process or cast her vote. In the case of testing, the vote does not count, the unveil information and the supposed content of the commitment is printed and the voting process begins anew. The voter can do the testing as often as she wishes. The printed commitments and corresponding unveil information can be checked later, either by the voter herself or by a competent person or institution of her choice. After the voter's real vote is cast, the last

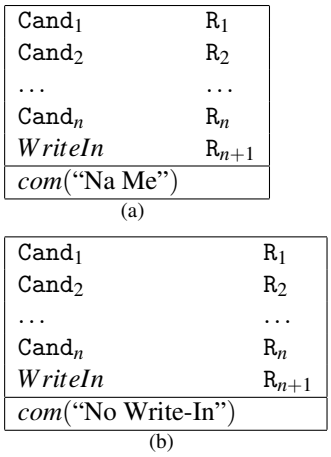

Figure 1: Scheme of a receipt with write-in support: 1(a) Receipt with write-in candidate "Na Me"; 1(b) No write-in candidate was chosen.

printed commitment remains unopened, and the voting process goes on as in the original scheme: The trusted random number generator creates the fresh random number representing the vote. Then the rest of the receipt is printed with the fresh random number behind the chosen candidate. If the voter voted for a write-in candidate, the fresh random number appears behind "Write-In". The voter can compare the random number behind her chosen candidate with the display of the trusted random number generator. Then she can take her receipt home to check later that her vote has been counted in the tally.

Please note that we do not have to make any additional assumptions about the printer. Since the dummy votes are printed last and do not have to be sent to the printer by the voting machine before, the voter can see what is printed anytime and take away the print-out anytime.

5.1.4 Post-voting Phase

After the election, the tally is published as sketched in Figure 2 together with all receipts and a proof of its correctness. As Figure 2 shows, the tally consists of two parts. The first part tallies the candidates on the list and yields the overall number of write-in votes. It is done as described in Section 4.1 by counting the unused dummy votes for each candidate. The proofs of correctness of this tally part are done as in the original scheme described in Section 4.1. The other part tallies the write-in candidates. To prove its correctness, we can use the same proof technique as for the first tally part. What has to be proven is that the outcome of the tally corresponds to the commitments behind "Write-In" on the receipts. The number of commitments to "No Write-In" has to equal the number of all votes minus the number of votes for write-in candidates.

Candidate	Number of votes
John	9
Joe	7
Write In	3

(a)

Write-in candidate	Number of votes
Alice	2
Bob	1
No Write-In	16

(b)

Figure 2: 2(a) Main tally: There are 3 votes for write-in candidates. 2(b) Tally of the write-in candidates: The number of "votes" for No Write-In has to match the number of non-write-in votes.

5.1.5 Some remarks

Our scheme provides the same amount of voter privacy as Bingo Voting with the exception that the abstention attack that forces the voter to write in a certain random string is still possible. This is a problem of all write-in techniques that publish the names of written-in candidates one-to-one in the tally. We address this problem in Section 5.2. An advantage of our scheme is that the voter's receipt neither shows a write-in candidate's name in plaintext nor if a candidate was written in at all.

There is good reason why we don't print the whole receipt before testing. If we would print the whole receipt at once, one had to consider what happens with the dummy votes if the voter chooses to test the voting process. If the voter saw the dummy random numbers, they cannot be reused on another receipt because the voter would recognize them. One cannot just use the same dummy votes on the voters real receipt because maybe she gives her vote to someone else. But if the dummy votes are thrown away, either an infinite pool of dummy votes has to be provided or the number of tests have to be limited, both possibilities are infeasible. Solution: The voter sees the dummy votes only after testing.

5.2 Bingo Voting with Write-In Support and Fuzziness

We now describe a μ,μ-fuzzy version of Bingo Voting with write-in support. To prevent forced abstention that is caused by forcing the voter to "vote" for a certain random string it is crucial here that the voter may write in names without voting for them, so that those names appear in the representation of the tally. Without this possibility, a voter can be forced to not vote for a certain write-in candidate by forcing him to write in another. So we have to provide the voter with the possibility to somehow whisper an arbitrary number of names to the voting authority that are later included in the representation even though they got no votes. Since in Bingo Voting for secrecy we have to assume an honest voting computer anyway we can as well provide the voting computer with an additional interface with which the voter can secretly enter names that she wishes to be included in the representation.

The scheme works as follows: Consider a μ,μ-fuzzable election with l eligible voters, one vote per voter, n regular candidates $C_1,\ldots,C_n$, and a set $\mathscr{R}$ of representations that is compliant to the representations in the post-voting phase described below.

5.2.1 Pre-Voting Phase

First, we add a candidate C_{n+1} called "Write In" to the candidate list. Then all candidates $C_1,\ldots,C_{n+1}$ get $l+\mu$ dummy votes as in the μ,μ-fuzzy Bingo Voting scheme without write-in support described in Section 4.2. The rest of this phase is also done as in Section 4.2, commitments to each candidate's dummy votes including "Write In" are created and a corresponding proof is published that each candidate has the same number of dummy votes.

5.2.2 Voting Phase

The voting phase is the same as in the non-fuzzy write-in supporting scheme described in Section 5.1 with the difference that the voter is additionally provided with an interface for entering names that she wishes to be included in the representation of the tally. Regardless of whether the voter entered a name using this interface or not, she can cast her vote by writing in a candidate or by voting for a regular candidate.

5.2.3 Post-Voting Phase

The post voting phase of this scheme is a combination of the μ,μ-fuzzy Bingo Voting version described in Section 4.2 and the write-in supporting scheme described in Section 5.1. Suppose that the voting phase yields a list of n_w shuffled additional names $N_1,\ldots,N_w$ that belong to write-in candidates and/or were entered via the interface in the voting booth, including "no write-in". These names can be real names or random strings. They should be in shuffled order because their order should yield no information about which candidates are actual write-in candidates and which names were entered via the additional interface. This list is published, and all the names in this list have to be included in the representation of the tally. Voters can check if the names they entered or voted for appear on the bulletin board.

Like in the write-in supporting scheme described in Section 5.1, we have two tally parts. We represent these

parts with two representation vectors. The tally

$$T^r = (T_1^r, \ldots, T_{n+1}^r)$$

of the regular candidates (including "Write-In") is represented by

$$R^r = (R_1^r, \ldots, R_{n+1}^r)$$

and the tally

$$T^w = (T_1^w, \ldots, T_{n_w}^w)$$

of the write-in candidates is represented by

$$R^w = (R_1^w, \ldots, R_{n_w}^w)$$

where R_i^w represents the number of votes for name N_i for all i.

The trusted voting authority first computes the tally T^r of the regular candidates. This tally is represented as in the μ,μ-fuzzy scheme described in Section 4.2 by choosing a vector $f = (f_1, \ldots, f_{n+1})$ with restrictions

1. $\sum_{i=1}^{n} f_i \geq \mu$ and
2. $0 \leq f_i \leq \min\{T_i, \mu\}$ for all i

and publicly opening $R_i^r = T_i^r + \mu - f_i$ dummy votes for each candidate C_i.

Then the voting authority opens the commitments to the write-in candidates in secret and computes the tally T^w. Another vector $f^w = (f_1^w, \ldots, f_{n_w}^w)$ is chosen with

1. $\sum_{i=1}^{n+1} f_i + \sum_{i=1}^{n_w} f_i^w \geq \mu$ and
2. $f_i^w \leq \min\{T_i^w, \mu\}$ for $i = 1, \ldots, n_w$.

Of each potential write-in candidate N_i, the voting authority publicly opens $R_i^w = T_i^w - f_i^w$ commitments, yielding the representation $R^w = (R_1^w, \ldots, R_{n_w}^w)$ of the tally of the write-in votes. We now have f_i unopened commitments for each regular candidate C_i and f_i^w unopened commitments for each write-in candidate N_i. Analogous to the μ,μ-fuzzy scheme described in Section 4.2 we fill this set of commitments so that there are μ commitments for each candidate. Those are opened via for example randomized partial checking, proving that each regular candidate C_i got between $R_i^r - \mu$ and R_i^r votes and each potential write-in candidate N_i got between R_i^w and $R_i^w + \mu$ votes. Please note that while we have μ additional dummy votes for each regular candidate, we don't have any extra commitments for the write-in candidates. This is why the representation of the write-in votes differs from the representation of the regular candidates.

5.2.4 Weak μ,μ-Fuzzy Scheme with Write-In Support

Analogous to the weak μ,μ-fuzzy scheme described in Section 4.3, the scheme described above can be made weak μ,μ-fuzzy by adapting the vectors f and f^w and not filling each candidate's commitments up to μ. If there are at least μ write-in votes and of the not written in candidates every candidate has at least μ votes, the first tally part can be published as it is. If necessary, the votes of the first tally part can be used to make the second tally part fuzzy though.

5.3 Some Remarks on Coercion-Resistance

To show how our definition of fuzziness can be used together with definitions of coercion resistance, we informally apply our work to the definition of Küsters et. al [18] and the μ,μ-fuzzy Bingo Voting schemes described in Section 4.2 and Section 5.2. Küsters et. al prove that Bingo Voting achieves the same degree of coercion resistance as an ideal voting scheme except for forced abstention. They state that Bingo Voting is vulnerable to this attack because the adversary sees the receipts of all voters. We argue that a voter can not by any feasible means prove that she has obtained no receipt, so she can not prove to an adversary that she did not vote. We do not regard this attack any further. What remains is forced abstention through voting for an unlikely or fictional candidate.

We informally define μ-coercion-resistance as follows:

> **Definition 7 (μ-coercion-resistance)** *A voting scheme applied to a μ,μ-fuzzable election is μ-coercion resistant if it is coercion-resistant according to the definition of Küsters et. al [18] and μ,μ-fuzzy.*

It is easy to see that if a μ,μ-fuzzy election scheme that achieves coercion resistance according to the definition of Küsters et. al, it also achieves the definition of μ-coercion-resistance above and is resistant against forced abstention caused by forcing to vote for unlikely candidates.

Since the μ,μ-fuzzy Bingo Voting scheme does not differ from the original scheme except in its published data and the fact that μ more dummy votes are created per candidate, and since its published data leaks less information than that of the original scheme, the μ,μ-fuzzy Bingo Voting should achieve coercion-resistance according to the definition of Küsters et. al. As stated above the voter cannot possibly prove that she did not vote by proving that she has no receipt. Therefore the scheme seems to be resistant against forced abstention.

The μ,μ-fuzzy Bingo Voting scheme with write-in support yields with its published data no more information as is yielded by an election done with the original scheme and all occurring write-in candidates on the list of regular candidates. The additionally published list of names does not tell if a candidate on this list got a write-in vote or not. So this scheme, too, should be μ-coercion-resistant and additionally resistant against forced abstention.

6 General Constructions

This section introduces a general construction of μ,μ-fuzzy verifiable election schemes from mix-based schemes and homomorphic schemes with a trusted authority. The constructions are similar to the construction of the μ,μ-fuzzy scheme from the original Bingo Voting scheme described in Section 4.2.

6.1 General Construction for Mixnet-Based Schemes with Trusted Entity

One assumption in the Bingo Voting schemes described in the previous sections is that there is a trusted entity that can open commitments in secret before publishing the representation of the tally and the proofs of correctness. One way to do the proofs of correctness in Bingo Voting is doing a mix with randomized partial checking. Other mix-based schemes can be made μ,μ-fuzzy in an analogous way, given that such a trusted entity is at hand. This is for example the case in the original Helios scheme [3], which also uses one trusted Helios server.

The construction is described for a verifiable mix-based election scheme applied to a μ,μ-fuzzable election with n candidates $C_1,\ldots,C_n$ and one vote per voter. We assume that there is a trusted entity E that gets all cast votes in the form of commitments or ciphertexts and has all the information needed to compute the plaintext votes. In the following, with *encrypted votes* we mean those ciphertexts or commitments. We also assume that more than μ votes are cast, so more than μ encrypted votes are available. The voting process of the original mix-based scheme stays the same up to the point where the tally is computed. Instead of mixing and opening the votes, we do the following. The trusted entity E opens the votes and computes the tally $T=(T_1,\ldots,T_n)$ in secret. Then E chooses $f=(f_1,\ldots,f_n)$ with

1. $\sum_{i=1}^{n} f_i \geq \mu$ and
2. $0 \leq f_i \leq \min\{\mu, T_i\}$ for all i.

Of the available encrypted votes the trusted entity takes away f_i from candidate C_i. The remaining $R_i = T_i - f_i$ encrypted votes of each candidate are included in the mix that would be done with all votes in the original scheme. The plaintexts of the votes included in the mix and a proof of correct mixing is published, again as would be done in the original scheme. The representation of the tally is $(R_1,\ldots,R_n) = (T_1 - f_1,\ldots,T_n - f_n)$. For each candidate C_i, $\mu - f_i$ new encrypted votes are created such that the corresponding plaintext votes are indistinguishable from the cast votes and each candidate has μ encrypted votes. Those $n*\mu$ encrypted votes are then mixed and opened with the same mix used for the representation of the tally, corresponding outcomes and proofs of correctness of the mix are published, proving that the tally lies between $(R_1,\ldots,R_n)$ and $(R_1+\mu,\ldots,R_n+\mu)$. This is different from the μ,μ-fuzzy Bingo Voting schemes because we don't have the extra μ commitments for each candidate.

A weak μ,μ-fuzzy election scheme can be created in the same way using corresponding vectors f and d as in the weak μ,μ-fuzzy Bingo Voting scheme described in Section 4.3.

Write-in support can be included analogous to the write-in supporting Bingo Voting scheme described in Section 5.2 using an additional "regular" candidate "Write-In" and two tallies with their own representations.

6.2 General Construction for Homomorphic Schemes with Trusted Entity

In this section, a verifiable μ,μ-fuzzy election scheme is constructed from an arbitrary verifiable election scheme with trusted entity that is based on homomorphic encryption. Analogous to the previous section, the construction is described for an election scheme applied to a μ,μ-fuzzable election with n candidates $C_1,\ldots,C_n$ and one vote per voter. Precondition is a trusted entity E who sees all encrypted votes and has the means to decrypt them. As in the previous section, the voting process of the μ,μ-fuzzy scheme is the same as in the original homomorphic scheme. After all votes are cast, the trusted entity E computes the tally $T=(T_1,\ldots,T_n)$ in secret. Then E chooses $f=(f_1,\ldots,f_n)$ with the usual restrictions:

1. $\sum_{i=1}^{n} f_i \geq \mu$ and
2. $0 \leq f_i \leq \min\{\mu, T_i\}$ for all i.

For each candidate C_i, E chooses f_i of his encrypted votes that are not to be included in the calculation of the sum of all encrypted votes. Then E calculates and publishes the sum of the remaining votes according to the original homomorphic scheme, and publishes which encryptions were not included in this calculation. The result is decrypted and the decryption proven as

would be in the original scheme, yielding a representation $R = (R_1, \ldots, R_n)$ of the tally where $R_i = T_i - f_i$. Then E creates $\mu - f_i$ encrypted votes for each candidate C_i whose corresponding plaintexts are indistinguishable from the votes cast by real voters. The sum of the encrypted votes not included in the calculation of the representation and the new encryptions is computed and decrypted, again with a proof of correct decryption. The result of the second sum should be μ votes for each candidate, which proves that each candidate C_i got between R_i and $R_i + \mu$ votes.

Here, too, a weak μ, μ-fuzzy version of the scheme can be created analogous to the construction above using corresponding vectors f and d as in the μ, μ-fuzzy Bingo Voting scheme described in Section 4.3.

Write-in support should also be possible to be included straightforwardly as described in Section 5.2.

7 Conclusion and Future Work

This work provides a formalization of fuzzy tally representations that closes the gap between coercion-resistance and write-in candidates. We have proposed a verifiable election scheme that supports write-in candidates and is resistant to forced-abstention attacks, as well as a general construction for such schemes. A remaining open problem is the construction of a verifiable μ, δ-fuzzy scheme that does not need a trusted authority.

Our notion of fuzziness is reminiscent of approaches to database anonymization like k-anonymity [11] and l-diversity [19]. Hence there seems to be a relation to database anonymity by modeling the tally as a database that is coarsened for privacy and which in its coarsened version must not leak any private information about the voter (namely her vote). Looking at database anonymization, we might want to have a tally representation that reaches something similar to Dwork's notion of differential privacy [13], but with end-to-end verifiability.

8 Acknowledgments

The author thanks Jörn Müller-Quade, Christian Henrich and the reviewers for helpful discussions and comments.

References

[1] ACQUISTI, A. Receipt-Free Homomorphic Elections and Write-in Ballots. Technical Report 2004/105, International Association for Cryptologic Research, 2004.

[2] ADIDA, B. Advances in Cryptographic Voting Systems. PhD Thesis, MIT, 2006.

[3] ADIDA, B. Helios: web-based open-audit voting. In *Proceedings of the 17th conference on Security symposium* (Berkeley, CA, USA, 2008), SS'08, USENIX Association, pp. 335–348.

[4] BENALOH, J. Simple verifiable elections. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop* (Berkeley, CA, USA, 2006), EVT'06, USENIX Association, pp. 5–5.

[5] BENALOH, J. Simple verifiable elections. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop* (Berkeley, CA, USA, 2006), USENIX Association, pp. 5–5.

[6] BOHLI, J.-M., HENRICH, C., KEMPKA, C., MÜLLER-QUADE, J., AND RÖHRICH, S. Enhancing Electronic Voting Machines on the Example of Bingo Voting. In *IEEE Transactions on Information Forensics and Security Vol. 4* (2009), pp. 745–750.

[7] BOHLI, J.-M., MÜLLER-QUADE, J., AND RÖHRICH, S. Bingo Voting: Secure and Coercion-Free Voting Using a Trusted Random Number Generator. In *VOTE-ID 2007* (2007), A. Alkassar and M. Volkamer, Eds., vol. 4896 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 111–124.

[8] CARBACK, R. T., CHAUM, D., CLARK, J., CONWAY, J., ESSEX, A., HERRNSON, P. S., MAYBERRY, T., POPOVENIUC, S., RIVEST, R. L., SHEN, E., SHERMAN, A. T., AND VORA, P. L. Scantegrity II Municipal Election at Takoma Park: The First E2E Binding Governmental Election with Ballot Privacy. Proceedings of the 19th USENIX Security Symposium, 2010.

[9] CHAUM, D., CARBACK, R., CLARK, J., ESSEX, A., POPOVENIUC, S., RIVEST, R. L., RYAN, P. Y. A., SHEN, E., AND SHERMAN, A. T. Scantegrity II: End-to-End Verifiability for Optical Scan Election Systems using Invisible Ink Confirmation , 2008. `http://www.usenix.org/event/evt08/tech/full_papers/chaum/chaum.pdf`.

[10] CHAUM, D., RYAN, P. Y., AND SCHNEIDER, S. A Practical Voter-Verifiable Election Scheme. In *Computer Security – ESORICS 2005* (2005), S. De Capitani di Vimercati, P. Syverson, and D. Gollmann, Eds., vol. 3679 of *Lecture Notes in Computer Science*, Springer, pp. 118–139.

[11] CIRIANI, V., DI VIMERCATI, S. D. C., FORESTI, S., AND SAMARATI, P. k-anonymity. In *Secure Data Management in Decentralized Systems*. 2007, pp. 323–353.

[12] DELAUNE, S., KREMER, S., RYAN, M. D., AND DELAUNE, S. Coercion-resistance and receipt-freeness in electronic voting. In *In Proc. 19th Computer Security Foundations Workshop* (2006), Press, pp. 28–42.

[13] DWORK, C. Differential privacy. In *in ICALP* (2006), Springer, pp. 1–12.

[14] GARDNER, R. W., GARERA, S., AND RUBIN, A. D. Coercion resistant end-to-end voting. In *Financial Cryptography* (2009), pp. 344–361.

[15] JAKOBSSON, M., JUELS, A., AND RIVEST, R. L. Making Mix Nets Robust For Electronic Voting By Randomized Partial Checking. In *USENIX Security Symposium* (2002), pp. 339–353.

[16] JUELS, A., CATALANO, D., AND JAKOBSSON, M. Coercion-resistant electronic elections. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society* (2005), ACM, pp. 61–70.

[17] KIAYIAS, A., AND YUNG, M. The Vector-Ballot E-Voting Approach. Financial Cryptography 2004, 2004.

[18] KÜSTERS, R., TRUDERUNG, T., AND VOGT, A. A Game-Based Definition of Coercion-Resistance and its Applications. *Journal of Computer Security (special issue of selected CSF 2010 papers)* (2011). To appear.

[19] MACHANAVAJJHALA, A., KIFER, D., GEHRKE, J., AND VENKITASUBRAMANIAM, M. L-diversity: Privacy beyond k-anonymity. *TKDD 1*, 1 (2007).

[20] MORAN, T., AND NAOR, M. Receipt-Free Universally-Verifiable Voting With Everlasting Privacy. In *Advances in Cryptology – CRYPTO 2006* (Aug. 2006), C. Dwork, Ed., vol. 4117 of *Lecture Notes in Computer Science*, Springer, pp. 373–392.

[21] NEFF, C. A. A verifiable secret shuffle and its application to e-voting. CCS '01 Proceedings of the 8th ACM conference on Computer and Communications Security, 2001.

[22] PEDERSEN, T. P. Non-interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Advances in Cryptology – CRYPTO '91: Proceedings* (1991), J. Feigenbaum, Ed., vol. 576 of *Lecture Notes in Computer Science*, Springer, pp. 129–140.

[23] POPOVENIUC, S., AND STANTON, J. Undervote and Pattern Voting: Vulnerability and a mitigation technique.

[24] TEAGUE, V., RAMCHEN, K., AND NAISH, L. Coercion-resistant tallying for stv voting. In *EVT* (2008).