Coercion-Resistant Electronic Elections

(Extended Abstract)

Ari Juels
RSA Laboratories
Bedford, MA, USA
ajuels@rsasecurity.com

Dario Catalano
CNRS - Ecole Normale Supérieure
75230 Paris Cedex 05
France
dario.catalano@ens.fr

Markus Jakobsson
Indiana University, School of Informatics
Bloomington, IN, USA
markus@indiana.edu

ABSTRACT

We introduce a model for electronic election schemes that involves a more powerful adversary than previous work. In particular, we allow the adversary to demand of coerced voters that they vote in a particular manner, abstain from voting, or even disclose their secret keys. We define a scheme to be *coercion-resistant* if it is infeasible for the adversary to determine if a coerced voter complies with the demands.

A first contribution of this paper is to describe and characterize a new and strengthened adversary for coercion in elections. (In doing so, we additionally present what we believe to be the first formal security definitions for electronic elections of *any* type.) A second contribution is to demonstrate a protocol that is secure against this adversary. While it is clear that a strengthening of attack models is of theoretical relevance, it is important to note that our results lie close to practicality. This is true both in that we model real-life threats (such as vote-buying and vote-canceling), and in that our proposed protocol combines a fair degree of efficiency with an unusual lack of structural complexity. Furthermore, previous schemes have required use of an untappable channel throughout. Ours only carries the much more practical requirement of an anonymous channel during the casting of ballots, and an untappable channel during registration (potentially using postal mail).

This extended abstract is a heavily truncated version of the full paper available at http://eprint.iacr.org/2002/165.

Categories and Subject Descriptors

H.m [**Information Systems**]: Miscellaneous; E.3 [**Data**]: Data Encryption

General Terms

Security

WPES '05, November 7, 2005, Alexandria, Virginia, USA.

Keywords

coercion-resistance, electronic voting, mix networks, receipt-freeness

1. INTRODUCTION

Most voters participating in shareholder elections in the United States have the option of casting ballots electronically via a Web browser [1]. Some voters near Geneva participating in recent referenda in Switzerland in 2003-4 have been able to cast binding votes over the Internet [19]. The UK government has enunciated plans to allow its citizens to cast votes electronically "some time after 2006" [18]. These are just a few instances of a broadening trend toward Internet-based voting. While voting of this kind appears to encourage higher voter turnout [38] and make accurate accounting for votes easier, it also carries the potential of making abuse easier to perform, and easier to perform at a large scale. A number of papers in the cryptographic literature have described ways of achieving robust and verifiable *electronic* elections, i.e., elections in which ballots and processing data are posted to a publicly accessible bulletin board. For just a few recent examples, see [8, 16, 21, 22, 27, 30, 34, 37, 40].

There are two other threats, however, that it is equally crucial to address in a fair and democratic election process: We speak of *voter coercion* and *vote buying*. Internet-based voting does not introduce these problems, but it does have the potential to exacerbate them by extending the reach and data collection abilities of an attacker. This is highlighted in one way by the presence of a notorious Web site that provides a forum for the auctioning of votes [2]. Seller compliance was in that case merely voluntary. Conventional Internet voting schemes, however, including those described in the literature, actually provide an attacker with ready-made tools for verifying voter behavior and thereby exerting influence or control over voters. Without careful system design, the threats of coercion and vote buying are potentially far more problematic in Internet voting schemes than in ordinary, physical voting schemes.

One commonly proposed way of achieving secure electronic voting systems is to use a cryptographic system known as a *mix network* [14]. This is a tool that enables a collection of servers to take as input a collection of ciphertexts and to output the corresponding plaintexts according to a secret permutation. A straightforward way to achieve an election system that preserves the privacy of voters, then, is

to assign a private digital signing key to each voter. To cast a ballot, the voter encrypts her choice and signs it, and then posts it to a bulletin board (i.e., a publicly accessible memory space). When all ballots have been collected and the corresponding signatures have been checked, the ciphertexts are passed through a mix network. The resulting plaintext versions of the voter choices may then be tallied. Thanks to the privacy preserving property of the mix network, an adversary cannot tell which vote was cast by which voter. This approach is frequently advocated in the mix-network literature, as in, e.g., [8, 14, 22, 27].

In an ordinary mix-based scheme of this kind, an adversary can coerce a voter straightforwardly. The adversary can simply furnish the voter with a ciphertext on a particular candidate, and then verify that the voter posted a ballot containing that ciphertext. Alternatively, the adversary can demand the private signing key of the voter and verify its correctness against the corresponding public key. An adversary attempting to buy votes can use the same means. Other types of cryptographic voting schemes, namely homomorphic schemes [5, 16] and schemes based on blind signatures [20, 37], suffer from similar vulnerabilities.

1.1 Previous work

Previous investigations of coercion-resistant voting have been confined to the property of *receipt-freeness*. Roughly stated, receipt-freeness is the inability of a voter to prove to an attacker that she voted in a particular manner, even if the voter wishes to do so. For a more formal definition, see [37]. The property of receipt-freeness ensures that an attacker cannot determine exact voter behavior and therefore cannot coerce a voter by dictating her choice of candidate. It also protects against vote-buying by preventing a potential vote buyer from obtaining proof of the behavior of voters; voters can thereby *pretend* to sell their votes, but defraud the vote buyer. The notion of receipt-freeness first appeared in work by Benaloh and Tuinstra [5]; their scheme, based on homomorphic encryption, was shown in [25] not to possess receipt-freeness as postulated. An independent introduction of the idea appeared in Niemi and Renvall [35]. Okamoto [36] proposed a voting scheme which he himself later showed to lack the postulated receipt-freeness; a repaired version by the same author, making use of blind signatures, appears in [37]. Sako and Kilian [39] proposed a multi-authority scheme employing a mix network to conceal candidate choices, and a homomorphic encryption scheme for production of the final tally. The modelling of their scheme was clarified and refined by Michels and Horster [33]. The Sako and Kilian scheme served as a conceptual basis for the later work of Hirt and Sako [25], followed by the more efficient approach of [3]; these two are the most efficient (and correct) receipt-free voting schemes to date. A recently proposed scheme by Magkos *et al.* [32] distinguishes itself by an approach relying on tamper-resistant hardware, but is flawed.[1]

All of these receipt-free voting schemes include impractical assumptions. For example, these schemes assume the availability of an *untappable channel* between the voter and the authorities, that is, a channel that provides perfect secrecy in an information-theoretic sense. (I.e., even encryption does not provide an untappable channel.) The scheme in [37] makes the even stronger assumption of an *anonymous* untappable channel. (It is also not very practical in that it requires voter interaction with the system three times in the course of an election.) Moreover, all of these schemes (excepting [37]) lose the property of coercion-resistance if the attacker can corrupt even one of the tallying authorities in a distributed setting. The scheme of Hirt and Sako still retains coercion-resistance when such corruption takes place, but only under the strong assumption that the voter knows *which* tallying authorities have been corrupted; the proposal of Baudron *et al.* has a similar property.

A still more serious problem with of all of the receipt-free voting schemes described in the literature, however, is the fact that the property of receipt-freeness alone fails to protect an election system against several forms of serious, real-world attack, which we enumerate here:

1. **Randomization attack** This attack was noted by Schoenmakers in 2000 [41]; he described its applicability to the scheme of Hirt and Sako. The idea is for an attacker to coerce a voter by requiring that she submit randomly composed balloting material. In this attack, the attacker (and perhaps even the voter) is unable to learn what candidate the voter cast a ballot for. The effect of the attack, however, is to nullify the choice of the voter with a large probability. For example, an attacker favoring the Republican party in a United States election would benefit from mounting a randomization attack against voters in a heavily Democratic district.

2. **Forced-abstention attack** This attack relates to the previous one based on randomization. The attacker here coerces a voter by demanding that she refrain from voting. All of the schemes cited above are vulnerable to this simple attack, because they authenticate voters directly in order to demonstrate that they are authorized to participate in the election. Thus, an attacker can see who has voted, and use this information to threaten and effectively bar voters from participation.[2]

3. **Simulation attack** The receipt-free schemes described above assume that the attacker cannot coerce a voter by causing her to divulge her private keying material after the registration process but prior to the election process. Such an attack, however, is a real and viable one in previous schemes, because they permit an attacker to verify the correctness of private keying ma-

[1] We are unaware of any mention of a break of this scheme in the literature, and therefore briefly describe one here. The Magkos *et al.* system employs an interactive honest-verifier ZK proof made by a smartcard to the voter. Presumably because of the simulability of this proof, the authors describe the proof as being "non-transferable." This is not true. An adversary can stipulate that the voter engage in the proof using a challenge that the adversary has pre-selected. The proof then becomes transferable, effectively a receipt for the adversary. As noted in [25], this type of attack also explains why *deniable encryption* [12] does not solve the problem of coercion in a voting system.

[2] An exception is the scheme in [37], which does not appear to be vulnerable to a forced-abstention attack. This is because the scheme seems to assume that the authority checks voter enrollment privately. In other words, the scheme does not permit public verification that participating voters are present on a published voter roll. This is potentially a problem in its own right.

terial. For example, in [37], the voter provides a digital signature which, if correct, results in the authority furnishing a blind digital signature. In [25], the voter, when casting a ballot, proves knowledge of a private key relative to a publicly committed value. In general, receipt-freeness does not prevent an attacker from coercing voters into divulging private keys or buying private keys from voters and then *simulating* these voters at will, i.e., voting on their behalf.

1.2 Our contribution

Our contribution in this paper is twofold. First, we investigate a stronger and broader notion of coercive attacks than receipt-freeness. This notion, which we refer to as *coercion-resistance*, captures what we believe to be the fullest possible range of adversarial behavior in a real-world, Internet-based voting scheme. A coercion-resistant scheme offers not only receipt-freeness, but also defense against randomization, forced-abstention, and simulation attacks – all potentially in the face of corruption of a minority of tallying authorities. We propose a formal definition of coercion-freeness in the body of this paper. Two other properties are essential for any voting scheme, whether or not it is coercion-resistant. These are *correctness* and *verifiability*. As formal definitions for these properties seem to be lacking in the literature, we provide them in the full paper; we thus provide what we believe to be the first formal security framework for electronic elections in general.

To demonstrate the practical realizability of our definitions, we describe a voting scheme that possesses the strong property of coercion-resistance proposed in this paper – and also the properties of correctness and verifiability. Our proposed scheme does not require untappable channels during the voting process, but instead assumes voter access to an anonymous channel at some point during the voting process. (Registration does involve an untappable channel, but may be achieved via, e.g., postal mail.) The assumption of anonymous channels in an election can be realized in a practical way. (Exactly how practical is subject to some debate.) For example, an adversary may be able to view communications on the Internet, but not trace IP addresses to identities effectively. Or voters may achieve anonymity by using public terminals in Internet cafés, libraries, workplaces, or physical polling places to participate in an otherwise electronic election. Popular use of mixnets would similarly help, e.g., [22, 34]; broadcast channels are another possible mechanism for anonymity. Anonymous channels are, of course, a strictly weaker assumption than untappable ones. Thus our scheme can make use of untappable channels too. The assumption of untappable channels may be reasonable in some limited cases: It is not a straightforward matter for an unsophisticated attacker to tap point-to-point communications on the Internet, for example.

Anonymous channels are in fact a minimal requirement for *any* coercion-resistant schemes: An attacker that can identify which voters have participated can obviously mount a forced-abstention attack.

A drawback of our scheme is that, even with use of asymptotically efficient mix networks as in [22, 34], the overhead for tallying authorities is quadratic in the number of voters. Thus the scheme is only practical for small elections. Our hope and belief, however, is that our proposed scheme might serve as the basis for refinements with a higher degree of practical application. We provide a security proof for our proposed scheme in the full paper.

1.3 Intuition behind our scheme

In a conventional voting scheme, and also in receipt-free schemes like [25], the voter V_i identifies herself at the time she casts her ballot. This may be accomplished by means of a digital signature on the ballot, or by an interactive authentication protocol. The key idea behind our scheme is for the identity of a voter to remain hidden during the election process, and for the validity of ballots instead to be checked blindly against a voter roll. When casting a ballot, a voter incorporates a concealed credential. This takes the form of a ciphertext on a secret value σ that is unique to the voter. The secret σ is a kind of *anonymous credential*, quite similar in spirit to, e.g., [9, 10]. To ensure that ballots are cast by legitimate voters, the tallying authority $\mathcal{T}$ performs a blind comparison between hidden credentials and a list $\vec{L}$ of encrypted credentials published by an election registrar $\mathcal{R}$ alongside the plaintext names of registered voters.

By means of mixing and blind comparison of ciphertext values, it is possible to check whether a concealed credential is in the list $\vec{L}$ or not, without revealing which voter the credential has been assigned to. In consequence, an attacker who is given a fake credential $\tilde{\sigma}$ by a coerced voter cannot tell whether or not the credential is valid. (The attacker will learn how many ballots were posted with bad credentials. Provided, however, that some spurious ones are injected by honest players, authorities, or even outsiders, the individuals associated with bad ballots will remain concealed.) Moreover, the attacker cannot mount randomization or forced-abstention attacks, since there is no feasible way to determine whether an individual voter has posted a ballot or not. In particular, after divulging fake credential $\tilde{\sigma}$, a voter can go and vote again using her real credential σ.

1.4 Organization

In section 2, we describe our setup and attack models and sketch a few of the major adversarial strategies. We provide formal definitions for the security property of coercion-resistance in section 3. We describe the particulars of our proposed scheme in section 4, prefaced by a summary of the underlying cryptographic building blocks. In the full paper, we offer formal definitions for the correctness and verifiability of election schemes, a detailed security-proof outline, and details on our choice of primitives for realizing our proposed scheme.

2. MODELLING

An election system consists of several sets of entities:

1. *Registrars:* Denoted by $\mathcal{R} = \{R_1, R_2, \ldots, R_{n_R}\}$, this is a set of n_R entities responsible for jointly issuing keying material, i.e., credentials to voters.

2. *Authorities (Talliers):* $\mathcal{T} = \{T_1, T_2, \ldots, T_{n_T}\}$ denotes the authorities responsible for processing ballots and jointly counting votes and publishing a final tally.

3. *Voters:* Denoted by $\mathcal{V} = \{V_1, V_2, \ldots, V_{n_V}\}$, the n_V voters are the entities participating in a given election administered by $\mathcal{R}$. We let index i be a public identifier for V_i.

We make use of a *bulletin board*, denoted by $\mathcal{BB}$. This is a piece of universally accessible memory to which all players have appendive-write access. In other words, any player can write data to $\mathcal{BB}$, but cannot overwrite or erase existing data. Moreover, voters will be able to read the contents of $\mathcal{BB}$ once the vote casting phase has ended. For notational convenience, we assume that data are written to $\mathcal{BB}$ in μ-bit blocks for an appropriate choice of μ. Shorter data segments may be padded appropriately. For simplicity of exposition, we assume no ordering on the contents of $\mathcal{BB}$.

2.1 Functions

We define a *candidate slate* $\vec{C}$ to be an ordered set of n_C distinct identifiers $\{c_1, c_2, \ldots, c_{n_C}\}$, each of which corresponds to a voter choice, typically a candidate or party name. In an election, choice c_j may be identified according to its index j. Thus, for cryptographic purposes the candidate slate consists of the integers $\{1, 2, \ldots, n_C\}$ and may be specified by n_C alone. We define a *tally* on an election under slate $\vec{C}$ to be a vector $\vec{X}$ of n_C positive integers $x_1, x_2, \ldots, x_{n_C}$ such that x_j indicates the number of votes cast for choice c_j. The functions composing an election system are then as follows:

- **Registering:** $\mathsf{register}(SK_{\mathcal{R}}, i, k_1) \rightarrow (sk_i, pk_i)$ takes as input the private registrar key $SK_{\mathcal{R}}$, a (voter) identifier i and a security parameter k_1, and outputs a key pair (sk_i, pk_i). This is computed jointly by players in $\mathcal{R}$, possibly in interaction with voter V_i.
- **Voting:** $\mathsf{vote}(sk, PK_{\mathcal{T}}, n_C, \beta, k_2) \rightarrow$ *ballot* takes as input a private voting key, the public key of the authorities $\mathcal{T}$, the candidate-slate specification n_C, a candidate selection β, and a security parameter k_2, and yields a ballot of bit length at most μ. The form of the ballot will vary depending on the design of the election system, but is in essence a digitally signed vote choice encrypted under $PK_{\mathcal{T}}$.
- **Tallying:** $\mathsf{tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3) \rightarrow (\vec{X}, P)$ takes as input the private key of the authority $\mathcal{T}$, the full contents of the bulletin board, the candidate-slate size, all public voting keys, and a security parameter k_3 and outputs a vote tally $\vec{X}$, along with a non-interactive proof P that the tally was correctly computed.
- **Verifying:** $\mathsf{verify}(PK_{\mathcal{T}}, \mathcal{BB}, n_C, \vec{X}, P) \rightarrow \{0, 1\}$ takes as input the public key of the authorities, the contents of the bulletin board, the candidate-slate size, the voting tally, and a non-interactive proof of correct tallying. It outputs a '0' if the tally is incorrect and a '1' otherwise. (We characterize the behavior of verify more formally in the full paper.)

We define an election scheme ES as the collection of these functions. Thus $\mathsf{ES} = \{\mathsf{register}, \mathsf{vote}, \mathsf{tally}, \mathsf{verify}\}$.

2.1.0.1 Remark.

There are many election models in use throughout the world. The model we propose here excludes important variants. In some systems, for example, voters are asked to rank candidate choices, rather than just listing those they favor. Many systems permit the use of *write-in* votes, i.e., the casting of a ballot in favor of a candidate not listed on the slate for the election. We exclude write-in voting from our model because it undermines the possibility of coercion resistance in any scheme where an observer can see a complete election tally including write-in votes. An attacker may, for example, require coerced voters to cast write-in ballots for candidate names consisting of random strings pre-specified by the attacker. This way, the attacker can: (1) Verify that coerced voters complied with instructions, by looking for the random strings the attacker furnished, and (2) Ensure that the votes of coerced voters are not counted, since random strings will most likely not correspond to real election choices. (Thus, this would combine the forced abstention attack and the randomization attack.)

2.2 Summary of the attack model

We consider the process for a single election as proceeding in these phases, corresponding largely with the functions enumerated in section 2.1:

1. **Setup:** Key pairs are generated for or by $\mathcal{R}$ and $\mathcal{T}$. The candidate slate $\vec{C}$ for the election is published by $\mathcal{R}$ with appropriate integrity protection.
2. **Registration:** The identities and eligibility of would-be voters are verified by $\mathcal{R}$. Given successful verification, an individual becomes a registered voter, receiving from $\mathcal{R}$ a credential permitting participation in the election. Previously registered voters may be able to re-use their credentials. $\mathcal{R}$ publishes a voter roll $\vec{L}$.
3. **Voting:** Referring to the candidate slate $\vec{C}$, registered voters use their credentials to cast ballots.
4. **Tallying:** The authority $\mathcal{T}$ processes the contents of the bulletin board $\mathcal{BB}$ so as to produce a tally vector $\vec{X}$ specifying the outcome of the election, along with a proof of correctness P of the tally.
5. **Verification:** Any player, not just voters, can refer to $\mathcal{BB}, P$ and $\vec{L}$ to verify the correctness of the tally produced by $\mathcal{T}$ in the previous phase.

2.2.1 Assumptions in setup phase

Our security definitions permit the possibility of static, active corruption by the adversary of a minority of players in $\mathcal{T}$ in the setup phase. The security of our construction then relies on generation of the key pair $(SK_{\mathcal{T}}, PK_{\mathcal{T}})$ by a trusted third party, or, alternatively, on an interactive, computationally secure key-generation protocol such as [24] between the players in $\mathcal{T}$.

2.2.2 Assumptions prior to registration

The adversary may coerce a voter prior to the registration phase, either requesting in advance that the voter retain transcripts of the registration process, or dictating voter interaction with the registrar in advance.

2.2.3 Assumptions in registration phase

We assume that this phase is trustworthy. Strong integrity of the registrar $\mathcal{R}$ is of course critical for any secure election system. To evade coercion, a voter must be able to receive a credential without adversarial interference. An adversary capable of corrupting and seizing the credentials of a voter in this initial phase can mount a full simulation attack. An

adversary capable of preventing a voter from registering can mount a forced-abstention attack.

We assume therefore that the voter receives her credential from $\mathcal{R}$ via an untappable channel. We are helped here by the fact that registration is generally an offline procedure. For example, a voter might receive her registration through the postal service. We must then assume that mail in the postal system is not subject to compromise. Alternatively, registration could be performed by voters in person.

It is further necessary to assume that an attacker cannot obtain registration transcript data after the fact. Where registration is electronic, for example, erasure of data from voter interaction with $\mathcal{R}$ must be compulsory by voters (e.g., enforced by smartcards). In a postal system, we must assume that the adversary cannot harvest registration letters from targeted voters (at least not many of them).

Finally, we must assume the integrity of the registration procedure on the inside, i.e., we must assume that $\mathcal{R}$ is trustworthy and does not leak credentials to the adversary.[3]

It may or may not be entirely feasible to achieve all of these assumptions around the registration process in real-world systems. We note, however, that if the registration process does *not* have the integrity we require here, then coercion-resistance is the least of our problems: Votes can then be tampered with.

2.2.4 Assumptions on voting, tallying and verification phases

Subsequent to the registration phase, we assume that the adversary may seize control of a minority of players in $\mathcal{T}$ and any number of voters in a static, active manner. (Since $\mathcal{R}$ does not participate in the process subsequent to registration, we need not consider corruption of $\mathcal{R}$ at this point.) The adversary may also attempt to coerce voters outside its control by requesting that they divulge private keying material[4] or behave in a prescribed manner in voting. Voters are assumed to be able to cast their ballots via anonymous channels, i.e., channels such that an attacker cannot determine whether or not a given voter cast a ballot. We noted some practical underpinnings for this assumption above. As also noted above, this assumption is a requirement for any election scheme to be fully coercion-resistant: If an attacker can tell whether or not a given voter cast a ballot, then the attacker can easily mount a forced-abstention attack. We assume that either sessions are integrity protected (on a per-session basis, since they are anonymous), or that the adversary has only passive interaction with the network.

2.2.5 Not considered here

We do not treat the problem of denial-of-service attacks. Mechanisms like [29] might help, but are beyond the scope of our research. We also do not treat the problem of enabling voters to verify that their votes have been counted. This is in principle possible whilst retaining coercion resistance. Honest voters could submit their credentials via anonymous channels for verification of proper processing by authorities. Authorities would concoct appropriate false replies for invalid credentials. We regard this as future work.

[3]Some relaxation of this assumption is possible. If $\mathcal{R}$ comprises multiple players, computations by $\mathcal{R}$ are distributed, and communication between players in $\mathcal{R}$ are made directly with voters, then even if a minority of players in $\mathcal{R}$ is corrupted, coercion-resistance is still possible. The catch is that voters must know *which* players in $\mathcal{R}$ have been corrupted. For details on this idea, see [25].

[4]We assume that the coercion takes place remotely. For example, the adversary may not continuously watch over the shoulder of a voter, monitor her hard-drive, etc. Our proposed protocol does potentially defend against some shoulder-surfing, however, by permitting voters to use fake keys and/or re-vote.

3. FORMAL DEFINITIONS

We now turn our attention to formal security definitions of the essential properties of *correctness*, *verifiability*, and *coercion-resistance*, respectively abbreviated *corr*, *ver*, and *c-resist*. Our definitions hinge on a set of experiments involving an adversary $\mathcal{A}$ in interaction with components of the election system ES. This adversary is assumed to retain state throughout the duration of an experiment. We formulate our experiments such that in all cases, the aim of the adversary is to cause an output value of '1'. Thus, for experiment $\mathbf{Exp}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot)$ on property $E \in (ver, corr, c\text{-}resist)$, we define $\mathbf{Succ}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \mathsf{Pr}[\mathbf{Exp}^{E}_{\mathsf{ES},\mathcal{A}}(\cdot) = \text{'1'}]$.

According to the standard definition, we say that a quantity $f(k)$ is *negligible* in k if for every positive integer c there is some l_c such that $f(k) < k^{-c}$ for $k > l_c$. In most cases, we use the term negligible alone to mean negligible with respect to the full set of relevant security parameters. Similarly, in saying that an algorithm has *polynomial running time*, we mean that its running time is asymptotically bounded by some polynomial in the relevant security parameters. As the properties of correctness and verifiability are of less relevance to our work than coercion-resistance, we relegate the first two definitions to the full paper.

3.0.6 Coercion resistance

Coercion resistance may be regarded as an extension of the basic property of privacy. Privacy in an election system is defined in terms of an adversary that cannot interact with voters during the election process. In particular, we say that an election is private if such an adversary cannot guess the vote of any voter better than an adversarial algorithm whose only input is the election tally. (Note, for example, in an election where all voters vote Republican, the system may have the property of privacy, even though the adversary knows how all voters cast their ballots in that election.)

Coercion resistance is a strong form of privacy in which it is assumed that the adversary may interact with voters. In particular, the adversary may instruct targeted voters to divulge their private keys subsequent to registration, or may specify that these voters cast ballots of a particular form. If the adversary can determine whether or not voters behaved as instructed, then the adversary is capable of blackmail or otherwise exercising undue influence over the election process. Hence a coercion-resistant voting system is one in which the user can deceive the adversary into thinking that she has behaved as instructed, when the voter has in fact cast a ballot according to her own intentions.

Our definition of coercion resistance requires addition of a new function to voting system ES:

- The function $\mathsf{fakekey}(PK_{\mathcal{T}}, sk, pk) \rightarrow \tilde{sk}$ takes as input the public key of the authorities, and the private/public key pair of the voter. It outputs a spurious key $\tilde{sk}$.

Of course, for the function fakekey to enable coercion resistance, the key $\tilde{sk}$ must be indistinguishable by the adversary $\mathcal{A}$ from a valid key, and only distinguishable by a majority of talliers $\mathcal{T}$. This property is captured in our experiment characterizing coercion resistance. To simplify the formulation of the experiment, we assume implicitly that tally is computed by an oracle (with knowledge of $SK_{\mathcal{T}}$). It suffices, however, for $\mathcal{T}$ to be computed via a protocol that achieves correct output and is computationally simulable by the adversary $\mathcal{A}$ (who, it will be recalled, may corrupt a minority of $\mathcal{T}$).

Our definition of coercion resistance centers on a kind of game between the adversary $\mathcal{A}$ and a voter targeted by the adversary for coercive attack. A coin is flipped; the outcome is represented by a bit b. If $b = 0$, then the voter $\mathcal{V}o$ casts a ballot of its choice β, and provides the adversary with a false voting key $\tilde{sk}$; in other words, the voter attempts to evade adversarial coercion. The voter here is modeled as a function $\mathcal{V}o$ that selects a ballot from the slate represented by n_C, or possibly a blank ballot ϕ. If $b = 1$, on the other hand, then the voter submits to the coercion of the adversary; she simply furnishes the adversary with her valid voting key sk, and does not cast a ballot. The task of the adversary is to guess the value of the coin b, that is, to determine whether or not the targeted voter in fact cast a ballot.

We characterize the voting pattern of honest voters in terms of a probability distribution D_{n,n_C}. This distribution models the state of knowledge of the adversary about the intentions of these voters. Of course, the adversary generally does not have perfect knowledge of the voting intentions of honest voters. (Indeed, as we explain below, imperfect adversarial knowledge is actually a requirement for meaningful coercion-resistance.) For a collection of n voters outside the control of the adversary – i.e., voters not subject to coercion – we characterize the view of the adversary in terms of a probability distribution D_{n,n_C}. We let ϕ be a symbol denoting a null ballot, i.e., an abstention, and let λ denote a ballot cast with an invalid credential. Then D_{n,n_C} is a distribution over vectors $(\beta_1, \beta_2, \ldots, \beta_n) \in (n_C \bigcup \phi \bigcup \lambda)^n$, i.e., over the set of possible ballot choices for an election plus abstentions and invalid ballots. For a set of n voting credentials $\{sk_i\}$, we let $\mathsf{vote}(\{sk_i\}, PK_{\mathcal{T}}, n_C, D_{n,n_C}, k_2)$ denote the casting of ballots according to distribution D_{n,n_C}. In other words, a vector $(\beta_1, \beta_2, \ldots, \beta_n)$ is drawn from D_{n,n_C} and vote β_i is cast using credential sk_i.

We are now ready to present an experiment *c-resist* that defines the game described above between an adversary and a voter targeted for coercion. Recall that k_1, k_2, and k_3 are security parameters defined above, n_V is the total number of eligible voters for the election, and n_C is the number of candidates, i.e., the size of the candidate slate. We let n_A denote the number of voters that may be completely controlled, i.e., corrupted by the adversary. We define $n_U = n_V - n_A - 1$. In other words, the number of uncertain votes n_U equals the total number of honest votes, i.e., the number of possible votes, minus those coming from voters controlled by the attacker, minus the vote coming from the voter the attacker is trying to coerce (in the experiment).

We consider a static adversary, i.e., one that selects voters to corrupt prior to protocol execution. We assume that the adversary has a list of "voter names," i.e., a roll of potential participating voters.

We let $\leftarrow$ denote assignment and $\Leftarrow$ denote the append operation, while % denotes the beginning of an annotative comment on the experiment. Our experiment treats the case in which the adversary seeks to coerce a single voter; extension of the definition to coercion of multiple voters is straightforward. The experiments defined here halt when an output value is produced. The full paper contains helpful comments on the pseudocode.

Experiment $\mathbf{Exp}^{c\text{-}resist}_{\mathsf{ES},\mathcal{A},H}(k_1, k_2, k_3, n_V, n_A, n_C)$

$V \leftarrow \mathcal{A}$(voter names, "control voters");
$\{(sk_i, pk_i) \leftarrow \mathsf{register}(SK_{\mathcal{R}}, i, k_2)\}_{i=1}^{n_V}$;
$j \leftarrow \mathcal{A}(\{sk_i\}_{i \in V}$, "set target voter");
$\beta \leftarrow \mathcal{V}o(n_C)$;
if $|V| \neq n_A$ or $j \notin \{1, 2, \ldots, n_V\} - V$ then
    output '0';
$b \in_U \{0, 1\}$;
if $b = 0$ then
    $\tilde{sk} \leftarrow \mathsf{fakekey}(PK_{\mathcal{T}}, sk_j, pk_j)$;
    $\mathcal{BB} \Leftarrow \mathsf{vote}(sk_j, PK_{\mathcal{T}}, n_C, \beta, k_2)$;
else
    $\tilde{sk} \leftarrow sk_j$;
$\mathcal{BB} \Leftarrow \mathsf{vote}(\{sk_i\}_{i \neq j, i \notin V}, PK_{\mathcal{T}}, n_C, D_{n_U, n_C}, k_2)$;
$\mathcal{BB} \Leftarrow \mathcal{A}(\tilde{sk}, \mathcal{BB}$, "cast ballots");
$(\vec{X}, P) \leftarrow \mathsf{tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3)$;
$b' \leftarrow \mathcal{A}(\vec{X}, P$, "guess b");
if $b' = b$ then
    output '1';
else
    output '0';

The adversary $\mathcal{A}$ in the above experiment is quite powerful, being capable (when $b = 1$) of complete coercion of the targeted voter. In order to characterize the success of $\mathcal{A}$, we must compare $\mathcal{A}$ with a second adversary $\mathcal{A}'$. $\mathcal{A}'$ is capable of coercion only within the framework of an ideal voting experiment *c-resist-ideal*. In other words, $\mathcal{A}'$ characterizes the type of security against coercion that we would like to achieve in ES.

The main feature we are aiming for in our ideal experiment *c-resist-ideal* is for $\mathcal{A}'$ to learn nothing from the private keys she acquires from corrupted players and from the coerced player. In particular, $\mathcal{A}'$ cannot use private keys to perform active attacks. We cause $\mathcal{A}'$ to express voting choices in a direct, ideal process; $\mathcal{A}'$ cannot cast ballots, but merely enumerates the choices of players in her control. Additionally, $\mathcal{A}$ cannot use private keys to learn information about the voting behavior of honest players or the coerced player. The *only* information that $\mathcal{A}'$ gets is the grand total $\vec{X}$ of votes in the election.

One feature of our experiment is counterintuitive. Because this is an ideal experiment, $\mathcal{A}'$ *is always* given $\tilde{sk}$ as the key of the coerced player. This is because $\mathcal{A}'$ should be unable to determine, on the basis of keying material, from the situation in which coercion is successful or unsuccessful.

We require a function for the definition. We include here an ideal function ideal-tally that tallies the ballots posted to $\mathcal{BB}$ in a special way. The function ideal-tally tallies in a normal manner all of the ballots cast by honest voters, i.e., prior to adversarial posting. The ballots cast by $\mathcal{A}'$, however, are treated specially. In particular, ideal-tally determines for each ballot B what the underlying private key

sk_i is. If $i \notin V$, i.e., if the private key is not one assigned to one of the corrupted players, then the corresponding vote is not counted. Additionally, any double vote is not counted, i.e., ideal-tally performs the weeding of double votes that normally occurs during the tallying procedure. Finally, ideal-tally does the following based on the value of the secret bit b. If $b = 0$, then ideal-tally does not count any ballot cast (by the adversary) using private key $\tilde{sk}$. If $b = 1$, then ideal-tally does include in the final tally a ballot cast using $\tilde{sk}$ (excluding double votes).

Our definition of ideal-tally here assumes that every ballot has a unique corresponding private key. This is true of most natural ballot structures (and true of our proposed scheme). This definition, of course, also assumes ideal functionality in ideal-tally, namely the ability to extract private keys and plaintext votes from ballots. We do not specify in our definition how this "oracle" power is achieved. In our proofs, we construct a simulator capable of performing this functionality required from ideal-tally.

Note that although $\mathcal{A}'$ receives the secret key of a coerced voter in our ideal experiment, this is really just a technical step. This secret key in fact provides $\mathcal{A}'$ with no information useful in voting, since the ideal function ideal-tally ensures against misuse of keys; also, this secret key can provide no information useful in learning votes, since $\mathcal{A}'$ never sees $\mathcal{BB}$.

We are now ready to present the experiment *c-resist-ideal* that characterizes the success of $\mathcal{A}'$.

Experiment $\mathbf{Exp}^{\textit{c-resist-ideal}}_{\mathsf{ES},\mathcal{A},H}(k_1, k_2, k_3, n_V, n_A, n_C)$
$V \leftarrow \mathcal{A}'$(voter names, "control voters");
$\{(sk_i, pk_i) \leftarrow \mathsf{register}(SK_{\mathcal{R}}, i, k_2)\}_{i=1}^{n_V}$;
$j \leftarrow \mathcal{A}'$("set target voter");
if $|V| \neq n_A$ or $j \notin \{1, 2, \ldots, n_V\} - V$ then
  output '0';
$\beta \leftarrow \mathcal{V}o(n_C)$;
$b \in_U \{0, 1\}$;
if $b = 0$ then
  $\mathcal{BB} \Leftarrow \mathsf{vote}(sk_j, PK_{\mathcal{T}}, n_C, \beta, k_2)$;
$\tilde{sk} \Leftarrow sk_j$;
$\mathcal{BB} \Leftarrow \mathsf{vote}(\{sk_i\}_{i\neq j, i\notin V}, PK_{\mathcal{T}}, n_C, D_{n_U,n_C}, k_2)$;
$\mathcal{BB} \Leftarrow \mathcal{A}'(\tilde{sk}, \{sk_i\}_{i\in V}$, "cast ballots");
$(\vec{X}, P) \leftarrow \mathsf{ideal\text{-}tally}(SK_{\mathcal{T}}, \mathcal{BB}, n_C, \{pk_i\}_{i=1}^{n_V}, k_3)$;
$b' \leftarrow \mathcal{A}(\vec{X}$, "guess b");
if $b' = b$ then
  output '1';
else
  output '0';

DEFINITION 1. *We define an election scheme* ES *as* coercion resistant *if for any polynomially-bounded adversary $\mathcal{A}$, any parameters n and n_C, and any probability distribution D_{n,n_C}, the quantity*

$$\mathbf{Adv}^{\textit{c-resist}}_{\mathsf{ES},\mathcal{A}} = \left|\mathbf{Succ}^{\textit{c-resist}}_{\mathsf{ES},\mathcal{A}}(\cdot) - \mathbf{Succ}^{\textit{c-resist-ideal}}_{\mathsf{ES},\mathcal{A}}(\cdot)\right|$$

is negligible in all security parameters for any voter function $\mathcal{V}o$.

Viewed intuitively, this definition means that in a real protocol execution, the adversary effectively learns nothing more than the election tally $\vec{X}$. The adversary cannot learn any significant information from the protocol execution itself, even when mounting an active attack.

3.1 The need for adversarial uncertainty

Even if an election scheme ES is coercion resistant by our definition, i.e., $|\mathbf{Succ}^{\textit{c-resist}}_{\mathsf{ES},\mathcal{A}}(\cdot) - \mathbf{Succ}^{\textit{c-resist-ideal}}_{\mathsf{ES},\mathcal{A}}(\cdot)|$ is negligibly close to 0, some (thankfully small) degree of coercion remains possible for many natural distributions D_{n,n_C}. In many settings, we may well have $\mathbf{Succ}^{\textit{c-resist-ideal}}_{\mathsf{ES},\mathcal{A}}(\cdot) \gg 0$. This seems counterintuitive, but in fact reflects a critical observation: The degree of possible coercion resistance is bounded below by adversarial uncertainty about the behavior of honest voters. This is true for *any* election scheme whatever.

For example, suppose that an adversary knows that a targeted voter aims to vote "Democrat" and that every other voter will vote "Republican." Then coercion is *unavoidable*. If a "Democrat" vote turns up, then the adversary will necessarily know that the targeted voter has succeeded in registering a vote. Similarly, if the adversary is attempting to coerce one voter in a given election and knows that all hundred of the other eligible voters will cast ballots, then the adversary can mount an abstention attack straightforwardly. The adversary in this case simply threatens the voter in the case that the total tally for the election is one hundred and one.

Viewed another way, coercion resistance depends on the "noise" or statistical uncertainty in the adversary's view of how honest voters will vote. In other words, it hinges on the distribution D_{n,n_C}. The more entropy in D_{n,n_C}, the better the level of attainable coercion resistance. D_{n,n_C} serves the purpose in our experiments of defining the distribution of the "noise" that conceals the behavior of voters targeted by the adversary for coercion.

To our benefit, it is natural to expect that in a real-world election an adversary can obtain only fragmentary knowledge about the likely behavior of voters. An adversary may know, for instance, that most of the voters in a given district will vote "Republican," but probably won't know exactly how many. This means that coercion-resistance is a viable possibility. (Additionally, it is possible for voting authorities – or indeed any entity – intentionally to inject "chaff" in the form of blank and invalid ballots into an election system.)

4. A COERCION-RESISTANT ELECTION PROTOCOL

We are now ready to introduce our protocol proposal. We begin by describing the cryptographic building blocks we employ. Where appropriate, we model these as ideal primitives, as discussed in the full paper.

4.0.1 Threshold public-key cryptosystem with re-encryption

Our first building block is a threshold public-key cryptosystem CS that permits re-encryption of ciphertexts with knowledge only of public parameters and keys. The private key for CS is held by $\mathcal{T}$ in our construction.

To describe our aim in the ideal, we would like any ciphertext E to be perfectly hiding. We would like decryption to be possible only by having a majority of players in $\mathcal{T}$ agree on a ciphertext to be decrypted. We model this latter ideal property as in terms of a special decryption oracle denoted

by $D\tilde{E}C$. We assume further that any decryption performed by $D\tilde{E}C$ is publicly verifiable.

4.0.2 Selected cryptosystem

At first El Gamal [23] may seem a natural choice of cryptosystem for our purposes. However this solution is not completely satisfying in our setting for the following reason.

Our construction will be proved to achieve coercion resistance under the *Decisional Diffie-Hellman assumption* (details of this proof are in the full paper). The basic idea of our proof is very simple: One assumes that there is an adversary $\mathcal{A}$ that has some advantage μ, in distinguish experiment *c-resist-real* from experiment *c-resist-ideal*, and constructs another algorithm $\mathcal{S}$ which, receives on input a challenge for the decisional Diffie-Hellman problem [7, 42] (g^a, g^b, g^c), and "using" $\mathcal{A}$, should be able to "solve" the challenge with a related advantage μ'. Now, as it will become apparent in section **??**, in order for our proof to go through correctly we require the simulator to be able to correctly decrypt a given ciphertext even when this ciphertext is constructed from the received challenge. Unfortunately basic El Gamal does not allow this. Indeed, if one encrypts a message m as $(g^a, g^c m)$ the simulator cannot decrypt simply because it does not know the discrete logarithm of g^c in base g^a.

To overcome this problem we adopt a modified version of the basic El Gamal scheme which can be seen as a simplified version of the well known Cramer-Shoup [17] cryptosystem (but only providing semantic security with respect to a passive adversary). We let $\mathcal{G}$ denote the algebraic group over which we employ this modified El Gamal (which we'll simply call M-El Gamal), and q denote the group order. For semantic security, we require that the Decision Diffie-Hellman assumption hold over $\mathcal{G}$. The public key is (g_1, g_2, h) where g_1, g_2, h are elements in $\mathcal{G}$. The secret key is $x_1, x_2 \in Z_q$ such that $h = g_1^{x_1} g_2^{x_2}$.

To encrypt m one computes $(A, B, C) = (g_1^r, g_2^r, h^r m)$ for random r. Decryption is similar to plain El Gamal: one computes $m = C/(A^{x_1} B^{x_2})$. One can easily verify (see [17]) that this variant allows correct decryption even when ciphertexts are constructed from the received challenge.

We let $\in_U$ here and elsewhere denote uniform, random selection from a set. A ciphertext in M-El Gamal on message $m \in \mathcal{G}$ takes the form $(\alpha, \beta, \gamma) = (mh^r, g_1^r, g_2^r)$ for $r \in_U Z_q$. For succinctness of notation, we sometimes let $E_h[m]$ denote a ciphertext on message m under public key h (assuming that g_1 and g_2 are considered public parameters).

Further details on the security of the scheme may be found in the full paper. An important feature of the M-El Gamal cryptosystem is that, exactly as the original version, it may be easily implemented in a threshold setting. In other words, the private keys x_1, x_2 may be distributed such that decryption can be performed by any quorum of share holders, without leakage of additional information. We exploit this distributed form of M-El Gamal in our proposed election scheme. As explained above, rather than focusing on a particular embodiment, we model the process by a decryption oracle denoted by $D\tilde{E}C$. We refer the reader to the full paper and to [13] for further discussion of threshold decryption in (plain) El Gamal.

4.0.3 Plaintext Equivalence Test (PET)

A *plaintext equivalence test* (PET) [26, 31] is cryptographic primitive that operates on ciphertexts in a threshold cryptosystem. The input to PET is a pair of ciphertexts; the output is a single bit indicating whether the corresponding plaintexts are equal or not. PET may be realized as an efficient distributed protocol that reveals no additional, non-negligible information about plaintexts. For a detailed description of efficient methods to perform this verification, along with security proofs, see [31]. Rather than focusing on a specific embodiment of PET, we model the ideal properties of the primitive as an oracle denoted by $P\tilde{E}T$, with the property of public verifiability.

4.0.4 Mix network

A (re-encryption) mix network (MN) is a distributed protocol that takes as input an ordered set $\vec{E} = \{E_1, E_2, \ldots, E_d\}$ of ciphertexts generated in a cryptosystem like El Gamal that permits re-encryption. The output of MN is an ordered set $\vec{E}' = \{E'_{\pi(1)}, E'_{\pi(2)}, \ldots, E'_{\pi(d)}\}$. Here, $E'_{\pi(i)}$ is a re-encryption of E_i, while π is a uniformly random, secret permutation. This is to say that MN randomly and secretly permutes and re-encrypts inputs. Thus, the special privacy property of a mix network is this: An adversary cannot determine which output ciphertext corresponds to which input ciphertext, i.e., which inputs and outputs have common plaintexts. Stated another way, an adversary cannot determine $\pi(j)$ for any j with probability non-negligibly better than a random guess. A number of good mix network constructions have been proposed that offer privacy and robustness against a static, active adversary capable of corrupting any minority of the n players (servers) performing the mix network operation, e.g., [22] and Neff [34]. These constructions can offer the additional property of *verifiability*. In other words, a proof is output that is checkable by any party and demonstrates, relative to $\vec{E}$ and the public key of the ciphertexts that $\vec{E}$ is correctly constructed. It is convenient to conceptualize MN as an ideal primitive in terms of an oracle $\tilde{MN}$ for MN with the property of public verifiability.

4.0.5 Proofs of knowledge

As sketched above, we use non-interactive proofs of knowledge [6] in a number of places. We do not describe these tools in detail, as they are standard in the literature. Instead, we refer the reader to, e.g. [15], for discussion of construction and logical composition of such protocols, and [11] for notation and discussion of efficient realization. As usual in the literature, our proofs involve instantiation of the random oracle model [4] in honest-verifier ZK proofs.

4.1 Our proposed protocol

4.1.1 Setup

The key pairs $(SK_{\mathcal{R}}, PK_{\mathcal{R}})$ and $(SK_{\mathcal{T}}, PK_{\mathcal{T}})$ are generated (in an appropriately trustworthy manner, as described above), and $PK_{\mathcal{T}}$ and $PK_{\mathcal{R}}$ are published along with all system parameters.

4.1.2 Registration

Upon proof of eligibility from V_i, the registrar $\mathcal{R}$ generates and transmits to V_i a random string $\sigma_i \in_U \mathcal{G}$, the voter credential. $\mathcal{R}$ then adds $S_i = E_{PK_{\mathcal{T}}}[\sigma_i]$ to the voter roll $\vec{L}$.[5]

[5]In our definitions above, we use the common terminology of private and public keys – with corresponding notation sk_i

The voter roll $\vec{L}$ is maintained on the bulletin board $\mathcal{BB}$ and digitally signed as appropriate by $\mathcal{R}$.

We assume the trustworthiness of $\mathcal{R}$ as explained above. Still, if desired, $\mathcal{R}$ can furnish V_i with a proof that S_i is a valid ciphertext on σ_i. Where erasure of voter secrets is not automatic, a *designated verifier proof* [28] is needed for coercion resistance.

4.1.3 Candidate-slate publication

$\mathcal{R}$ or some other authority publishes an integrity-protected candidate slate $\vec{C}$ with names and unique identifiers in $\mathcal{G}$ for n_C candidates. This authority also publishes a unique, random election identifier ϵ.

4.1.4 Voting

Voter V_i casts a ballot for candidate c_j comprising M-El Gamal ciphertexts $(E_1^{(i)}, E_2^{(i)})$ respectively on choice c_j and credential σ_i. In particular, for $a_1, a_2 \in_U Z_q$:

$$E_1^{(i)} = (\alpha_1, \alpha_1', \beta_1) = (g_1^{a_1}, g_2^{a_1}, c_j h^{a_1}) \text{ and}$$

$$E_2^{(i)} = (\alpha_2, \alpha_2', \beta_2) = (g_1^{a_2}, g_2^{a_2}, \sigma_i h^{a_2}).$$

The first is a ciphertext on the candidate choice of the voter, the second a ciphertext on the credential of the voter.

Additionally, V_i includes non-interactive proofs of knowledge of σ_i and c_j, and also a proof that $c_j \in \vec{C}$, i.e., that c_j represents a valid candidate choice. The latter can be accomplished, for example, using a disjunctive proof that the ciphertext constitutes a valid encryption of a candidate choice in $\vec{C}$. It is needed because an invalid candidate choice is like a write-in: It can effectively serve as a receipt. These proofs, which we denote collectively by Pf, may be accomplished efficiently using standard techniques. As is standard practice, the challenge values for Pf derive from calls to a cryptographic hash function, modeled in our security analysis by a random oracle $O\tilde{W}$. Input to $O\tilde{W}$ for these challenge values includes ϵ, E_1, E_2 and commitment values required for the non-interactive proofs of knowledge. V_i posts $B_i = (E_1, E_2, Pf)$ to $\mathcal{BB}$ via an anonymous channel.

Note: This is not a receipt! By assumption, the adversary does not know whether or not a given voter has even posted a vote. The adversary therefore can't simply coerce by requesting decryption information.

4.1.5 Tallying

To tally the ballots posted to $\mathcal{BB}$, the authority $\mathcal{T}$ performs the following steps:

1. **Checking proofs:** $\mathcal{T}$ verifies the correctness of all proofs on $\mathcal{BB}$. Any ballots with invalid proofs are discarded. For the valid, remaining ballots, let $\vec{A}_1$ denote the list of ciphertexts on candidate choices (i.e., the E_1 ciphertexts), and let $\vec{B}_1$ denote the list of ciphertexts on credentials (i.e., the E_2 ciphertexts).

2. **Eliminating duplicates:** The tallying authority $\mathcal{T}$ performs pairwise PETs on all ciphertexts in $\vec{B}_1$, and removes duplicates according to some pre-determined policy, e.g., order of postings to $\mathcal{BB}$. When an element is removed from $\vec{B}_1$, the corresponding element (i.e., that with the same index) is removed from $\vec{A}_1$. We let $\vec{B}_1'$ and $\vec{A}_1'$ be the resulting "weeded" vectors. This process retains at most one ballot per credential.

3. **Mixing:** $\mathcal{T}$ applies MN to $\vec{A}_1'$ and $\vec{B}_1'$ (using the same, secret permutation for both). Let $\vec{A}_2$ and $\vec{B}_2$ be the resulting lists of ciphertexts.

4. **Checking credentials:** $\mathcal{T}$ applies mix network MN to the encrypted list $\vec{L}$ of credentials from the voter roll. $\mathcal{T}$ then compares each ciphertext of $\vec{B}_2$ to the ciphertexts of $\vec{L}$ using PET. $\mathcal{T}$ retains a vector $\vec{A}_3$ of all ciphertexts of $\vec{A}_2$ for which the corresponding elements of $\vec{B}_2$ match an element of $\vec{L}$ according to PET. This step achieves the weeding of ballots based on invalid voter credentials.

5. **Tallying:** $\mathcal{T}$ decrypts all ciphertexts in $\vec{A}_3$ and tallies the final result.

4.1.6 How to cheat a coercer

One possible implementation of the function fakekey is simply for the coerced voter V_i to claim that a random group element $\tilde{\sigma}_i$ is the true credential σ_i. (If coerced multiple times, the voter V_i releases the same value $\tilde{\sigma}_i$.) We discuss faking of voting keys in more detail in the full paper.

and pk_i – to describe the credentials associated with voters. Shifting from a general exposition to our specific protocol, we now use σ_i instead of sk_i to denote a voter credential, and S_i instead of pk_i to denote a public representation thereof. This notational change reflects voters' use of an unconventional form of public-key authentication in our scheme.

5. REFERENCES

[1] Proxyvote.com: Shareholder election website, 2005. URL: www.proxyvote.com.

[2] Vote-auction, 2005. URL: www.vote-auction.net.

[3] O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, and G. Poupard. Practical multi-candidate election system. In *PODC 2001*, pages 274–283. ACM Press, 2001.

[4] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

[5] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *26th ACM STOC*, pages 544–553, 1994.

[6] M. Blum, A. D. Santis, S. Micali, and G. Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, 1991.

[7] D. Boneh. The Decision Diffie-Hellman problem. In *ANTS '98*, pages 48–63. Springer-Verlag, 1998. LNCS no. 1423.

[8] D. Boneh and P. Golle. Almost entirely correct mixing with applications to voting. In V. Atluri, editor, *ACM CCS '02*, pages 68–77. ACM Press, 2002.

[9] S. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy.* MIT Press, 2000.

[10] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT '01*, pages 93–118. Springer-Verlag, 2001. LNCS no. 2045.

[11] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In B. Kaliski, editor, *CRYPTO '97*, pages 410–424. Springer-Verlag, 1997. LNCS no. 1294.

[12] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In B. Kaliski, editor, *CRYPTO '97*, pages 90–104, 1997. LNCS no. 1294.

[13] R. Canetti, R. Gennaro, S. J. H. Krawczyk, and T. Rabin. Adaptive security for threshold cryptosystems. In M. Wiener, editor, *CRYPTO '99*, pages 98–115. Springer-Verlag, 1999. LNCS no. 1666.

[14] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[15] R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. Desmedt, editor, *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994. LNCS no. 839.

[16] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In W. Fumy, editor, *EUROCRYPT '97*, pages 103–118. Springer-Verlag, 1997. LNCS no. 1233.

[17] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO '98*, pages 13–25. Springer-Verlag, 1998. LNCS no. 1462.

[18] M. Cross. Public domain. *The Guardian: Guardian Unlimited Online*, 10 June 2004.

[19] J. Fowler. Switzerland tests virtual democracy in national referendum. *Technology Review*, 26 September 2004.

[20] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In J. Seberry and Y. Zheng, editors, *ASIACRYPT '92*, pages 244–251. Springer-Verlag, 1992. LNCS no. 718.

[21] J. Furukawa. Efficient, verifiable shuffle decryption and its requirement of unlinkability. In B. et al., editor, *PKC 04*, pages 319–332. Springer-Verlag, 2004. LNCS no. 2947.

[22] J. Furukawa and K. Sako. An efficient scheme for proving a shuffle. In J. Kilian, editor, *CRYPTO '01*, volume 2139 of *Lecture Notes in Computer Science*, pages 368–387. Springer-Verlag, 2001.

[23] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[24] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. The (in)security of distributed key generation in dlog-based cryptosystems. In J. Stern, editor, *EUROCRYPT '99*, pages 295–310. Springer-Verlag, 1999. LNCS no. 1592.

[25] M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In B. Preneel, editor, *EUROCRYPT '00*, pages 539–556, 2000. LNCS no. 1807.

[26] M. Jakobsson and A. Juels. Mix and match: Secure function evaluation via ciphertexts. In T. Okamoto, editor, *Advances in Cryptology - Asiacrypt '00*, pages 162–177. Springer-Verlag, 2000. LNCS No. 1976.

[27] M. Jakobsson, A. Juels, and R. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In D. Boneh, editor, *USENIX '02*, pages 339–353, 2002.

[28] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In U. Maurer, editor, *EUROCRYPT '96*, pages 143–154. Springer-Verlag, 1996. LNCS no. 1070.

[29] A. Juels and J. Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In *Proceedings of the 1999 ISOC Network and Distributed System Security Symposium*, pages 151–165, 1999.

[30] A. Kiayias and M. Yung. Self-tallying elections and perfect ballot secrecy. In D. Naccache and P. Paillier, editors, *PKC '02*, pages 141–158. Springer-Verlag, 2000. LNCS no. 2274.

[31] P. MacKenzie, T. Shrimpton, and M. Jakobsson. Threshold password-authenticated key exchange. In M. Yung, editor, *CRYPTO '02*, pages 385–400, 2002. LNCS no. 2442.

[32] E. Magkos, M. Burmester, and V. Chrissikopoulos. Receipt-freeness in large-scale elections without untappable channels. In B. S. *et al.*, editor, *First IFIP Conference on E-Commerce, E-Business, E-Government (I3E)*, pages 683–694, 2001.

[33] M. Michels and P. Horster. Some remarks on a receipt-free and universally verifiable mix-type voting scheme. In K. Kim and T. Matsumoto, editors, *ASIACRYPT '96*. Springer-Verlag, 1996. LNCS no. 1163.

[34] A. Neff. A verifiable secret shuffle and its application to e-voting. In P. Samarati, editor, *ACM CCS '01*, pages 116–125. ACM Press, 2001.

[35] V. Niemi and A. Renvall. How to prevent buying of votes in computer elections. In J. Pieprzyk and R. Safavi-Naini, editors, *ASIACRYPT '94*, pages 164–170. Springer-Verlag, 1994. LNCS no. 917.

[36] T. Okamoto. An electronic voting scheme. In N. T. *et al.*, editor, *IFIP World Congress*, pages 21–30, 1996.

[37] T. Okamoto. Receipt-free electronic voting schemes for large scale elections. In B. C. *et al.*, editor, *Security Protocols Workshop*, pages 25–35. Springer-Verlag, 1997. LNCS no. 1361.

[38] S. Parker. Shaking voter apathy up with IT. *The Guardian*, 11 Dec. 2001.

[39] K. Sako and J. Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In L. Guillou and J.-J. Quisquater, editors, *EUROCRYPT '95*, pages 393–403. Springer-Verlag, 1995. LNCS no. 921.

[40] B. Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In M. Wiener, editor, *CRYPTO '99*, pages 148–164. Springer-Verlag, 1999. LNCS no. 1666.

[41] B. Schoenmakers, 2000. Personal communication.

[42] Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In *Workshop on Practice and Theory in Public Key Cryptography (PKC '98)*. Springer, 1998.