Remote Electronic Voting with Revocable Anonymity

Matt Smart and Eike Ritter

School of Computer Science
University of Birmingham, UK
{m.j.smart, e.ritter}@cs.bham.ac.uk

Abstract. We present a new remote, coercion-free electronic voting protocol which satisfies a number of properties previously considered contradictory. We introduce (and justify) the idea of revocable anonymity in electronic voting, on the grounds of it being a legal requirement in the United Kingdom, and show a method of proving the validity of a ballot to a verifier in zero knowledge, by extension of known two-candidate proofs.

1 Introduction

It is undoubtedly a challenge to design electronic voting systems that satisfy what is an ever-growing list of requirements that are difficult to achieve simultaneously. Many governments have begun to adopt electronic voting with a view to improving voter turnout, with hardly any success. One of the driving factors for electronic voting is *remote voting*—the requirement that a citizen can vote from any location. This is unfortunately very difficult to achieve whilst minimising the potential for voter coercion. Further, how is it possible to satisfy voter privacy (anonymity) whilst also allowing the voter to verify that her vote has been counted? Can we assure ballot correctness for any number of candidates?

An especially important property in electronic voting is *anonymity* (privacy)—the notion that no voter should be linkable to their ballot. In this work, we introduce *revocable anonymity* to electronic voting — the notion that it should be possible to link one’s identity to one’s ballot, but *only* with the agreement of a Judge and a quorum of mutually distrusting parties.

In the UK, it is a legal requirement that it should be possible for the election authorities to link a ballot to its voter [2, p. 106]. To our knowledge, no work has previously considered this notion (equivalent to revocable anonymity in electronic voting), but it seems important to do so (note that some other protocols may be able to achieve this with modification [16, 6], but give no detail as to how to do so, and would seemingly not provide a sufficient solution—we discuss this later). One can envisage a situation in which, since voters are entirely anonymous, an attacker can vote on behalf of people who should be unable to vote, but, for whatever reason, are still on the electoral roll. In a 2005 postal vote scandal in Birmingham, UK, “possibly well over 2,000 of the votes cast”

were fraudulent and illegitimate for one ward alone [26]. We feel that permitting anonymity revocation in extreme circumstances is fundamental to reducing election fraud such as this.

1.1 Related Work

In our experience, there is no work which provides revocable anonymity in electronic voting, and little work which provides large-scale, coercion-resistant, remote electronic voting ([6] is a good example, but does not seem scalable). We here discuss how previous authors have satisfied some of the important properties of e-voting without revocable anonymity, and our strategy to achieve them.

Many electronic voting protocols [24, 10, 16, 9, 28] rely on anonymous channels, or anonymous and untappable channels [19], to satisfy some security properties. When considering voting over the Internet (an inherently insecure medium), one needs to think about how an anonymous channel could be implemented in the first place.

Attempts have been made to achieve anonymous channels with mix networks [4, 22, 18, 24, 3, 13], which provide effective anonymity, but can often be slow, inefficient, complex and subject to single points of failure (in the case of decryption mixes). Indeed, it has been argued [27] that for an Internet-based voting protocol, there is no way to reliably implement an anonymous communication channel over the Internet. Volkamer and Krimmer [27] suggest that IP address tracking or trojan horse viruses alone mean that any attempt at an anonymous channel would always suffer from some weakness.

Thankfully, in our work, we do not need to use anonymous *or* untappable channels (which are, when from voter to talliers, a very strong assumption), relying instead on various designated verifier proofs to satisfy voter verifiability whilst maintaining coercion-resistance and privacy.

In our work, we follow the scheme of many previous protocols using homomorphic encryption to ensure universal verifiability and unlinkability of ballots [1, 7, 8, 11, 28, 16], which naturally lends itself to threshold cryptography, affording us a greater level of assurance against corrupted talliers. These protocols, along with some of those already mentioned, require, for remote voting, that the voter is not observed at the “very moment of voting” [18]. Indeed, Benaloh and Tuinstra state that “physical separation of the voter from possible coercive agents is fundamental to any uncoercible election protocol” [1, p. 550].

Lee *et al.* [18], amongst others, suggest the use of a *tamper-resistant randomiser*—smart card—and non-voter-observation at the point of voting, to guarantee coercion-resistance. An alternative is to have every voter use a public voting booth which either uses a smart card, as above, or a paper ballot which is optically processed by machine [4, 22, 21, 5, 20].

We note that any protocol providing a list of voters’ identities with encrypted ballots could provide revocable anonymity, given the collusion of all parties needed to perform decryption. However, such a list clearly violates full coercion-resistance, as the fact that a voter has voted successfully can be determined by anyone. Juels *et al.* [16] and implementations thereof [6] involve talliers only keeping a list of votes at the end of the election (discarding the pre-

vious stage’s encrypted credentials), thus severing the direct link between voter and vote. Revocation of anonymity would require a highly inefficient Plaintext Equivalence Test between the credential supplied with a vote and every credential on the voter list, followed by a collusion with the registrar. Lee *et al.* [18] would allow for revocation, *but* subject to collusion of the administrator, the entire mix and n talliers. The nature of usage of the bulletin board in the protocol also suggests that full coercion-resistance is not possible, as the fact that Alice has voted is plainly visible. Prêt à Voter [22] and similar schemes do not offer revocation at all, since Alice’s choice of ballot paper is random, and as any identifying information is destroyed (by Alice), she cannot be linked to her ballot. In any case, no other protocol discusses revocable anonymity at all, to our knowledge. We note that revocable anonymity is a concept which has been considered at great length in other fields, such as digital cash [14, 15, 17].

In digital cash, it is particularly important that it should be possible to link an electronic coin to the person who spent it once the transaction has occurred (for example, that coin may have been spent twice, or spent illegally). It is similarly important to be able to link a person’s identity to all coins available to him (for example, to protect against money laundering). One manner in which this can be done is to encode an encrypted copy of the coin owner’s identity into every coin. Requiring two or more parties to perform encryption, including a judge [15], ensures that a user’s anonymity won’t be revoked unless there is sufficient legal cause. In our work, we protect the voter’s identity using a similar mechanism.

1.2 Our Contribution

In this work, we introduce a remote electronic voting protocol which satisfies several properties considered important in electronic voting, leading to several contributions:

- A secure voting protocol allowing a quorum of authorities to link a ballot to its voter (revocable anonymity), whilst achieving coercion-resistance and legitimate voter privacy
- A novel method of allowing the voter to achieve coercion-resistance without anonymous channels or tamper-resistant devices, through designated-verifier signatures
- An extension of previous schemes to prove ballot validity for two-candidate elections, to multiple-candidate elections

The protocol we present achieves the above properties, as well as the standard electronic voting properties (completeness, uniqueness, coercion-resistance, fairness, and legitimate-voter privacy), while having no need for anonymous or untappable channels (or implementations thereof).

Protocol Schema We present a two-phase protocol, where voters do not need to synchronise between phases they are actively involved in. Our reasoning for splitting into two phases is to preserve the anonymity of the legitimate voter, henceforth referred to as Alice. In the first phase, voters receive eligibility tokens

with designated verifier signatures, and form ElGamal encryptions of ballots, submitting them to a bulletin board. A member of a semi-trusted tallier group re-encrypts Alice’s vote.

In the second phase, Alice receives a designated verifier proof of re-encryption (along with some other fake proofs), and her re-encrypted vote is posted to another bulletin board with an encrypted version of her identity. Alice can then check her vote has been included, or contact a **Judge** otherwise.

Once all votes are posted to the second bulletin board, a tally is calculated and announced. A simple schematic diagram of the protocol is given in Figure 1.

Fig. 1. A schematic for our protocol.

1.3 Structure

In §2, we define a number of preliminaries, including the terminology used, and a number of primitives which we make use of. In §3, we give the participants, trust model and threat model for our work. We present our protocol in §4, and the requirements we have satisfied in §5. Finally, we conclude.

2 Preliminaries

In this paper, we assume the availability of the following cryptographic primitives. Note that we are working in the formal model, not in provable security. Therefore we make the assumption that the cryptography in the primitives below is perfect.

2.1 Threshold ElGamal Encryption Scheme

We use a standard ElGamal encryption scheme under a q -order multiplicative subgroup $G_q = \langle g \rangle$ of $\mathbb{Z}_p^*$, generated by an element $g \in \mathbb{Z}_p^*$, where p and q are suitably large primes, and $q|(p-1)$. All agents a in the protocol have a private key s_a of which only they have knowledge. Each agent has a corresponding public key $h_a = g^{s_a}$ where g is a known generator of the subgroup. Public keys are common knowledge to all users. For more information on the encryption scheme we use, the reader is directed to the appropriate paper. We use a (t, n) -threshold decryption scheme analogous to that of Cramer *et al.* [8]. For brevity we do not discuss this here.

2.2 Strong Designated Verifier Signature Scheme

We adopt the designated verifier signature scheme of Saeednia *et al.* [23] due to its efficient nature, but others would be acceptable. We use designated verifier signatures to enable a prover (Bob, or any one of the first-round talliers in our case) to prove a statement to a verifier (Alice) by proving the validity of a signature. However, Alice is unable to prove the signature's validity to *anyone* else, on the grounds that she could have produced it herself [23, p. 43]. For brevity we do not discuss the scheme here, but direct the reader to the appropriate paper instead [23].

2.3 Proof of Equality of Discrete Logarithms

In order to prevent an attack in our voting scheme (voting for several candidates or for one candidate multiple times with the same ballot), we require that the voter demonstrates to a verifier that her vote is of the correct form (without revealing what the vote is).

As we discuss later, a voter's vote is of the form $(x, y) = (g^\alpha, h_{\mathbb{T}_2}^\alpha g^{M^{i-1}})$ where $\alpha \in_R \mathbb{Z}_q$, M is the maximum number of voters and i represents the position in the list of candidates of the voter's chosen candidate. Alice needs to prove, in zero knowledge, that she is sending to the bulletin board some value for y where the exponent of g is in $\{M^0, \dots, M^{L-1}\}$ where L is the number of candidates. If we did not have such a proof, any voter could spoil the election by adding spurious coefficients to the exponent, thereby voting several times.

We extend the technique of Cramer *et al.* [8], who use a non-interactive proof of equality of discrete logarithms to prove the validity of a ballot in a *two*-candidate election. We extend the two-candidate scenario to L candidates, providing a proof for the relation given by

$$\log_g x = \log_{h_{\mathbb{T}_2}}(y/g^{M^0}) \vee \dots \vee \log_g x = \log_{h_{\mathbb{T}_2}}(y/g^{M^{L-1}})$$

In Figure 2, we give a generalised adaptation (**G-PEQDL**) of the above proof of equality of discrete logarithms scheme where Alice votes for candidate k ($1 \leq k \leq L$) with $(x, y) = (g^\alpha, h^\alpha g^{M^{k-1}})$. This is the only place where we extend one of the primitives we use. We provide a more detailed explanation (and proof) of the G-PEQDL in [25].

2.4 Designated Verifier Re-encryption Proofs

The properties of the ElGamal encryption scheme allow re-encryption (randomisation) of ciphertexts. Given a ciphertext (x, y) , another agent is able to generate a re-encryption $(x_f, y_f) = (xg^\beta, yh^\beta)$, where $\beta \in_R \mathbb{Z}_q^*$.

In our protocol, we use an ElGamal re-encryption to preserve the voter's anonymity. However, the voter needs to have some conviction that her vote has been counted (individual verifiability). We achieve this via a *Designated Verifier Re-encryption Proof* (DVRP) based on Alice's keypair: such a proof convinces Alice that a given re-encrypted ciphertext is equivalent to that she generated, whilst not convincing any third party¹. We adopt the scheme used by Lee *et*

¹ Note that in order to fully protect against Alice's private key being stolen from her, we could give her, during in-person registration, a new public/private keypair

Fig. 2. Our generalised non-interactive proof of ballot validity for a vote for candidate k

al. [18, 12], such that the prover, P (the agent that does the re-encryption) demonstrates to Alice that (x_f, y_f) is equivalent to (x, y) in such a manner that the original message m (encrypted in (x, y)) is not revealed, and this proof cannot convince any other entity. The reader is directed to the appropriate papers for more details.

3 Protocol Model

3.1 Participants

Our protocol is modelled with 5 kinds of participants. A participant (agent) is an interactive polynomial-time random computation. All agents are able to communicate via a network, which is not secure or anonymous in any way.

The participants are as follows:

- **Voters.** The protocol allows M voters $v_i \in \{v_0, v_1, \dots, v_{M-1}\}$ to vote. Alice is an honest voter who wishes to vote anonymously. She is able to vote many times, but *once* unobserved. Eligible voters’ public keys are publicly known.
- **First Round Bulletin Board/First Round Talliers.** Our protocol uses two separate *bulletin boards*. A standard bulletin board is a public broadcast channel with memory. The first bulletin board we use is writable only by *voters*. All voters send an encrypted vote and signed proof of validity to this board, which we denote as $\mathcal{BB}_1$.

The *first-round talliers* $\mathbb{T}_1$ are a *semi-trusted* group of agents², each possessing an ElGamal secret key $s_{\mathbb{T}_1}$ in its entirety, which any one of them can

$(s_{\text{Alice}-v}, h_{\text{Alice}-v})$, which acts as a session key for Alice’s vote. This pair could then be used for the DVRP. However, such a modification is not strictly necessary.

² We discuss our need for trusting $\mathbb{T}_1$ later in this Section.

use to remove the first layer of encryption on Alice’s vote³. We assume that each instance would be busy enough, and that votes would be batched before sending to $\mathcal{BB}_2$, so that timing attacks would be ineffective. Our justification for having multiple members of $\mathbb{T}_1$ is to prevent a bottleneck of computational power, but if this problem were ignored, we could equally substitute the group for a single entity.

The first round talliers are responsible for ensuring that Alice’s vote is valid according to the set of valid possible votes, not coerced, and not a double-vote. They are unable to see Alice’s actual vote token. $\mathbb{T}_1$ also encrypts Alice’s identity, should anonymity revocation be required. They issue Alice with vote validity tokens during registration.

- **Second Round Bulletin Board/Second Round Talliers.** The second bulletin board $\mathcal{BB}_2$ is viewable by all users of the protocol, and writable only by $\mathbb{T}_1$. It lists only the re-encrypted (valid) votes in a random permutation. The votes themselves, (x, y) , are encrypted with the public key of the second round talliers.

The *second-round talliers* are a group of agents (disjoint from $\mathbb{T}_1$) who decrypt the ballots listed on the second round bulletin board using threshold ElGamal with a shared key $s_{\mathbb{T}_2}$. The second round talliers will also publish the final tally.

- **Anonymity Teller Group.** As well as each being separate groups $\mathbb{T}_1, \mathbb{T}_2$, the tallier groups form part of a larger group which deals only with the voter’s anonymity. This group contains an equal number of members of $\mathbb{T}_1$ and $\mathbb{T}_2$ and is simply denoted $\mathbb{T}$. As such, it has a public key $g^{s_{\mathbb{T}}}$ and associated private key $s_{\mathbb{T}}$, where the private key is distributed amongst all members as before. In this case, to decrypt, a quorum of a size t_{id} , greater than the size of either $\mathbb{T}_1$ or $\mathbb{T}_2$, will need to collude to decrypt. Note that this decryption is only ever needed when a voter’s identity needs to be traced, as our protocol is optimistic. Further, a voter’s anonymity cannot be revoked without the agreement of the quorum *and* the Judge.
- **Judge.** The Judge is an entity of the protocol that is rarely used. She has two purposes:
 1. If Alice cannot find her re-encrypted vote on the bulletin board, she asks the Judge for verification.
 2. The Judge also authorises anonymity revocation (having been presented with appropriate evidence of the need for revocation) in order to deliberately link a ballot to a voter, by applying her private key for a decryption.

Note that the Judge is only used in a minority of cases, i.e., where a voter’s identity needs to be revealed, or Alice cannot find her vote on the bulletin board. The Judge, understandably, is trusted. We note that she could equally be formed from a coalition of mutually distrusting parties, disjoint from

³ The size of $\mathbb{T}_1$ would need to be determined empirically depending on the size of the electorate. Since each member of the group has a copy of the same key, the size only affects how much of a bottleneck (in terms of computational power) $\mathbb{T}_1$ is.

$\mathbb{T}_1/\mathbb{T}_2$, and selected by the electoral authorities. However, we see the **Judge** more in terms of a physical arbiter of justice in a court of law.

(Partially) Trusting $\mathbb{T}_1$ The purpose of the first-round talliers is to check the eligibility of Alice to vote and to re-encrypt Alice’s vote before it is posted to the second bulletin board. To achieve anonymity, we need to *partially* trust $\mathbb{T}_1$. This means that we trust that $\mathbb{T}_1$:

- will not reveal the link between Alice’s ballot (x, y) and her re-encrypted ballot (x_f, y_f) , except by request of the **Judge**;
- will make valid encryptions of voter identities when forming $\overline{\text{id}}$ tags;
- will act honestly in communications with the **Judge** (no other honest communications are required than those stated here);
- will only sign and post to $\mathcal{BB}_2$ ballots which are valid

Note therefore that $\mathbb{T}_1$ *at no point* has access to Alice’s unencrypted vote. We further do not trust $\mathbb{T}_1$ to reliably send communications—if messages do not arrive as expected, the voter can detect this.

We believe that the trust we have placed in $\mathbb{T}_1$ is the minimum assumption necessary to assure the properties we wish to satisfy. We further discuss our decision in [25].

3.2 Trust Model

We make the following assumptions in our protocol:

1. All parties trust that $\mathbb{T}_1$ will not reveal the link between a ballot (x, y) and its re-encryption (x_f, y_f)
2. All parties trust that $\mathbb{T}_1$ will perform valid encryptions of each voter’s identity, to afford anonymity revocation
3. The **Judge** and $\mathbb{T}_2$ trust that $\mathbb{T}_1$ will only sign and post to $\mathcal{BB}_2$ ballots which are valid
4. The **Judge** trusts that $\mathbb{T}_1$ will accurately and honestly send any data requested by it, to the **Judge**
5. All participants trust that the **Judge** will only authorise revocation of anonymity in appropriate circumstances
6. Alice trusts that she will receive one (and only one) valid voting token, along with several invalid ones, from the first-round talliers during registration.
7. Alice trusts the **Judge** to honestly state whether votes have been counted
8. All parties trust that voter identities will be stored correctly (and securely) on the second-round bulletin board

Note that we have already assumed that: $\mathbb{T}_1$ will batch votes before sending to $\mathcal{BB}_2$, to prevent timing attacks; Alice can vote once unobserved; and a t -sized quorum of $\mathbb{T}_2$ will not collude to break fairness or decrypt ballots until voting is over.

3.3 Threat Model

In this section, we consider the potential threats that could affect our protocol, based on the attacker's capabilities. We address how these threats are managed in §4. As to the assumptions we make about the attacker's strength based on the strength of the cryptography we use, we assume perfect cryptography.

Note that in our protocol, the attacker can assume the role of any entity (except the Judge). He is able to corrupt up to $t - 1$ talliers where collusion is required to decrypt messages (and t is the threshold size for that quorum). All channels are public, so the attacker can:

1. Read messages
2. Decrypt and read any message m , subject to having the correct decryption key s for an encrypted message $(g^\alpha, g^{\alpha s}m)$
3. Intercept messages
4. Inject bad ballots in the first phase, and spurious messages generally
5. Temporarily block messages (although we assume resilient channels for liveness)

4 Protocol

Our voting protocol has four stages:

Fig. 3. Our protocol. Dashed lines indicate a non-compulsory part of the protocol (complaints). Note that the first communication ($T_1 \rightarrow$ Alice) is in-person.

Stage 1: Ballot Validity Tokens

The protocol begins with Alice registering *in person* to vote (this would be with T_1). At this point, she receives a *random number of* values δ_i , which are

generated *at the point of registration*. Each has a designated verifier signature $\text{DVSign}_{\mathbb{T}_1}(\delta_i)$ paired with it, which has been generated by a member of $\mathbb{T}_1$. However, only one of these signatures is valid (clearly, only the voter with the correct private key can verify this fact⁴). Alice hence receives a string

$$\langle (\delta_0, \text{DVSign}_{\mathbb{T}_1}(\delta_0)), (\delta_1, \text{DVSign}_{\mathbb{T}_1}(\delta_1)), \dots, (\delta_{n-1}, \text{DVSign}_{\mathbb{T}_1}(\delta_{n-1})) \rangle$$

The coercion-resistance Alice enjoys increases with $|n|$ (i.e., the probability that the attacker can guess the correct δ value decreases with $|n|$).

Note that Alice would be able to generate designated verifier signatures at her liberty. Alice is able to calculate which of the signatures is valid for the value paired with it, and the tallier stores, on a private electoral roll (accessible only to $\mathbb{T}_1$) the valid δ value for Alice with her name. If Alice votes under coercion, since she received a random number of δ values, an observer cannot force her to use all values (she could conceal one or more, or arbitrarily insert values). Hence she simply votes using invalid δ values.

If she later votes without coercion⁵, she sends the correct δ value with her vote as a ‘proof’ of validity. Upon checking for eligibility, the talliers simply check Alice’s submitted δ value against the correct one stored on the private electoral roll. If she were to send a value for which the DV-Signature was incorrect when sent to her, this would alert the first-round talliers that her vote was made under coercion, which would alter their response to her. However, a coercer would not be able to distinguish a valid δ value from an invalid one, as he has no way of determining whether Alice herself made the designated verifier signature, or indeed whether the signature is valid.

Stage 2: Encrypted Vote Posting

As with other voting protocols using homomorphic encryption, we choose the form of the ballot in such a way that decryption of all ballots multiplied together leads to a simple tally of votes. A vote for the i^{th} candidate is given as $g^{M^{i-1}}$, where M is the maximum number of voters.

Voter Alice selects a value $\alpha \in_R \mathbb{Z}_q$, and encrypts her vote for candidate i using the public key of the *second round talliers*, to give $(x, y) = (g^\alpha, h_{\mathbb{T}_2}^\alpha g^{M^{i-1}})$. She groups this with the correct δ value δ_A , and her public key h_{Alice} . Finally, she calculates the Generalised Proof of Equality of Discrete Logarithms (see §2.3) for her ballot (x, y) to prove that the vote is of correct form, and produces a standard ElGamal signature on this. This tuple $\langle (x, y), \text{Sign}_{\text{Alice}}(\mathbf{G}\text{-PEQDL}), \delta_A, h_{\text{Alice}} \rangle$ is encrypted with the public key of the first-round talliers, and posted to the first round bulletin board, $\mathcal{BB}_1$.

Stage 3: Validity Checking

Once Stage 2 is complete, any member $\mathbb{T}_{1_i}$ of $\mathbb{T}_1$ removes the first layer of encryption on each vote on the first-round bulletin board. That tallier then:

⁴ Note that in Figure 3, Alice also receives a token `vcid`. In practice, she would remember this value, and use it to demonstrate to her computer that she was not being coerced, rather than having to verify designated verifier signatures herself.

⁵ We assume that Alice is able to vote unobserved, but she only needs to do this once.

1. verifies that the vote is legitimate, by ensuring that the δ value given is the one stored with Alice's name on the private electoral roll⁶. Note that because the votes themselves are encrypted for $\mathbb{T}_2$, the first-round talliers cannot see *how* a voter votes — merely *that* a voter has attempted to vote.
2. verifies the G-PEQDL supplied with the ballot (x, y) to determine that Alice's vote is a single vote for a single valid candidate in the election

Once the validity of a ballot is assured, and any invalid ballots are disposed of, $\mathbb{T}_{1_i}$ re-encrypts (x, y) with a random factor β to give (x_f, y_f) . That member also encrypts Alice's public key by doing the following:

- Select a random $\phi \in_R \mathbb{Z}_q$
- Using the joint public key for both sets of talliers $h_{\mathbb{T}}$, and the Judge's public key, form $\bar{\text{id}} = (g^\phi, h_{\mathbb{T}}^\phi h_{\text{Judge}}^\phi h_{\text{Alice}})$.

The tallier then continues. He:

3. generates a signature on $\text{hash}(x_f, y_f, \bar{\text{id}})$, and concatenates this with $(x_f, y_f, \bar{\text{id}})$ to form the final message string.

The tallier responsible for the re-encryption sends Alice a designated-verifier re-encryption proof (DVRP) that her vote has been included on the public bulletin board as (x_f, y_f) , along with a number of other correct DVRPs, which are not valid for Alice (only she will be able to determine this). Note that if Alice's sent δ value were invalid, the tallier would send Alice only invalid DVRPs, meaning that an attacker could not determine whether her vote was invalid simply by observing messages received by Alice. As before, Alice would be free to insert seemingly valid DVRPs into the communication. The tallier also sends Alice a signature of her original vote, $\text{sign}_{\mathbb{T}_1}(x, y)$.

The tallier will then personally store the values $\langle (x, y), \beta \rangle$, and mark on the private electoral roll that Alice has voted (for example, by adding a signature of her public key). This information will never be released, except to the Judge as proof that Alice's vote was counted. The tuple $\langle x_f, y_f, \bar{\text{id}}, \text{sign}_{\mathbb{T}_1}(\text{hash}(x_f, y_f, \bar{\text{id}})) \rangle$ is posted to the second-round talliers' bulletin board. Alice is able to check the second bulletin board to ensure her vote appears and the signature on it is valid, but cannot convince anyone else of this fact (nor can she decrypt the re-encrypted vote). Any entity can check that a vote on the bulletin board is valid by verifying the signature for the hash of that vote.

Stage 4: Tallying

Once all DVRPs have been sent to their respective voters, it is simple for the second-round talliers $\mathbb{T}_2$ to decrypt votes. First, each $\langle (x_f, y_f), \bar{\text{id}} \rangle$ is checked against its signed hash. Those not matching are ignored in tallying. A quorum of t talliers jointly decrypt a product

$$(X, Y) = \left(\prod_{j=1}^l x_{f_j}, \prod_{j=1}^l y_{f_j} \right)$$

⁶ We presume that the private electoral roll is made inaccessible (or unconvincing) to coercers. We could accomplish this with designated verifier signatures.

(without any single member having access to the private key, as discussed in §2.1), and then post the product to a publicly viewable place. The quorum threshold-decrypt the resulting tally, giving $g^{r_1 M^0 + r_2 M^1 + \dots + r_L M^{L-1}}$, and $r_1, \dots, r_L$ as the final tally. Note that any party can verify that any vote must have been correct, by comparing each published hash to the values given with it.

Anonymity Revocation

We have built into our protocol the ability to recover a voter's identity after the voting process is complete, but only with the co-operation of the **Judge** and a quorum of $\mathbb{T}$, the anonymity group. When Alice's vote is submitted to $\mathcal{BB}_2$, part of it is a token $\overline{\text{id}} = (g^\phi, h_{\mathbb{T}}^\phi h_{\text{Judge}}^\phi h_{\text{Alice}})$. If, in the tallying phase of the protocol, any ballot is found to be illegal (or if, for any other reason, anonymity has to be revoked), a quorum of members of the *anonymity tallier group* $\mathbb{T}$ need to collude (note that the t_{id} value for this threshold decryption should be higher than the size of either $\mathbb{T}_1$ or $\mathbb{T}_2$).

$$\frac{h_{\mathbb{T}}^\phi h_{\text{Judge}}^\phi h_{\text{Alice}}}{g^{\phi s_{\mathbb{T}}}} = h_{\text{Judge}}^\phi h_{\text{Alice}}$$

The **Judge** must now be sent the token, with appropriate evidence justifying anonymity revocation. The **Judge** can then divide by $g^{\phi s_{\text{Judge}}}$ to give the voter's identity.

Voter Complaints

A disadvantage of using designated-verifier re-encryption proofs is that Alice cannot prove the validity of the proof she receives from the first-round talliers that her vote has been re-encrypted as (x_f, y_f) , which she may need to do if she cannot find her re-encrypted vote on $\mathcal{BB}_2$.

A solution we might adopt would be for Alice to receive a 1-out-of- L re-encryption proof [12], which is requested by Alice after all votes are posted to the board. However, such a proof is quite laborious and would allow an attacker to see that Alice's vote was counted. Instead, Alice sends her original (x, y) to the **Judge**, along with $\text{sign}_{\mathbb{T}_1}(x, y)$ as proof that she did indeed submit that vote. The **Judge** requests the stored β from the first-round talliers, and can then use these to check that Alice's vote was counted. If Alice's vote is counted, the **Judge** sends her a designated verifier signature for her public key, h_{Alice} . Otherwise, she makes the designated verifier signature invalid. Only Alice can determine this fact, and can again insert valid signatures arbitrarily. If Alice's vote is shown to have not been counted, we could also allow her to collude with the **Judge** to submit a vote a second time—in this manner, if her vote is again not counted, the **Judge** can take further action.

5 Properties of the Protocol

We now present the properties which our protocol satisfies. The explanations of how we satisfy each property are beyond the scope this paper and are provided in [25]. We use the Dolev-Yao model and hence assume that the cryptographic

operations presented in §2 are perfect; in other words the intruder is not able to break any of these cryptographic schemes but is able to intercept, change and delete all messages. We assume resilient channels to obtain liveness properties.

1. **Eligibility** Only eligible voters should be able to vote.
2. **Uniqueness** Only one vote per voter should be counted
3. **Receipt-Freeness** The voter should be given no information which can be used to demonstrate to a coercer how *or if* they have voted, *after* voting has occurred
4. **Coercion-Resistance** It should not be possible for a voter to prove how they voted or even if they are voting, even if they are able to interact with the coercer during voting
5. **Verifiability**
 - (a) **Individual Verifiability** A voter should be able to verify that their vote has been counted correctly
 - (b) **Universal Verifiability** Any observer should be able to verify that all votes have been counted correctly
6. **Fairness** No-one can gain any information about the result of the tally until the end of the voting process and publication of votes
7. **Vote Privacy** Neither the authorities nor any other participant should be able to link any ballot to the voter having cast it, *unless* the protocol to revoke anonymity has been invoked
 - (a) **Revocable Anonymity** It should be possible for an *authorised entity* (or collaboration of entities, for us) to reveal the identity of any *single* voter by linking his vote to him.
8. **Remote Voting** Voters should not be restricted by physical location

It should be noted that even in the event that $\mathbb{T}_1$ were not trusted and became compromised, vote privacy, fairness, and individual verifiability (in so much that Alice can ensure her vote is counted), are still satisfied—these are not dependent on trusting $\mathbb{T}_1$. Receipt-freeness and coercion-resistance are satisfied in that Alice still cannot show *how* she votes.

The assumptions we make on $\mathbb{T}_1$ make it unnecessary to require assumptions made in other approaches on remote electronic voting, e.g. anonymous, often untappable channels [24, 9–11, 7], availability of a trusted Smart-Card or ‘randomiser’ to perform re-encryptions and proofs thereof [18, 11, 9], or the assumption that the voter cannot be observed at all during voting. It should be noted that using a Smart-Card to re-encrypt instead of $\mathbb{T}_1$ would affect other properties, such as eligibility and remote voting.

6 Conclusion

We have presented an election protocol providing what we believe to be the first scheme for revocable anonymity, whilst also allowing the voter coercion-free remote voting and verifiability, as well as legitimate-voter privacy. We require no untappable channels, and achieve an efficient 1-out-of- L scheme, integrating an extension of two-candidate discrete proofs of logarithm equality to L parties.

Our protocol also satisfies *remote voting*: as long as the voter is connected to the Internet, they are able to vote from any location, under the assumption that they can vote unobserved once, and register in person at any point before the election. This, we argue, is a very natural assumption to make.

We envisage that, given the remote nature of our protocol, it could be implemented across the Internet. Future work will concentrate on removing the need to trust any single party (except the **Judge**) at all, and on enhancing the remote-voting nature of the protocol—we might consider how to ensure that the machine the voter votes from can be trusted by the voter. One option might be to allow the user to vote only through a signed applet.

6.1 Acknowledgements

We would like to thank Mark Ryan and Tom Chothia for their helpful comments on earlier versions of this paper.

References

1. Josh Benaloh and Dwight Tuinstra. Receipt-Free Secret-Ballot Elections (Extended Abstract). In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 544–53, Montreal, 1994. ACM.
2. Robert Blackburn. *The Electoral System in Britain*. Macmillan, London, 1995.
3. Dan Boneh and Philippe Golle. Almost Entirely Correct Mixing with Applications to Voting. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 68–77, Washington DC, 2002. ACM.
4. David Chaum, Peter Y. A. Ryan, and Steve Schneider. A Practical, Voter-verifiable Election Scheme. In *Computer Security — ESORICS 2005 10th European Symposium on Research in Computer Security*, volume 3679 of *Lecture Notes in Computer Science*, pages 118–39, Milan, September 2005. Springer-Verlag.
5. David Chaum, Jeroen van de Graaf, Peter Y. A. Ryan, and Poorvi L. Vora. High Integrity Elections. Cryptology ePrint Archive, Report 2007/270, 2007. <http://eprint.iacr.org/>.
6. Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a Secure Voting System. In *Proceedings, 2008 IEEE Symposium on Security and Privacy*, pages 354–68. IEEE, 2008.
7. Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. Multi-Authority Secret-Ballot Elections with Linear Work. In *Advances in Cryptology - EUROCRYPT'96. Proceedings*, pages 72–83, Berlin, 1996. Springer-Verlag.
8. Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In *Advances in Cryptology - EUROCRYPT '97. Proceedings*, pages 103–18, Berlin, 1997. Springer-Verlag.
9. Chun-I Fan and Wei-Zhe Sun. An efficient multi-receipt mechanism for uncoercible anonymous electronic voting. *Mathematical and Computer Modelling*, 48:1611–1627, 2008.
10. Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A Practical Secret voting Scheme for Large Scale Elections. In *Advances in Cryptology - AUSCRYPT '92. Workshop on the Theory and Application of Cryptographic Techniques Proceedings*, pages 244–51, Berlin, 1993. Springer-Verlag.
11. Martin Hirt. *Multi-Party Computation: Efficient Protocols, General Adversaries and Voting*. PhD thesis, ETH Zurich, 2001.
12. Martin Hirt and Kazue Sako. Efficient Receipt-Free Voting Based on Homomorphic Encryption. In *Advances in Cryptology - EUROCRYPT 2000. Proceedings*, volume

1807 of *Lecture Notes in Computer Science*, pages 539–56, Bruges, Belgium, 2000. Springer-Verlag.

13. Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making Mix Nets Robust for Electronic Voting by Randomised Partial Checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–53, Berkeley, 2002. USENIX Assoc.
14. Markus Jakobsson, David M’Raihi, Yiannis Tsionis, and Moti Yung. Electronic Payments: Where Do We Go From Here? In R. Baumgard, editor, *Secure Networking - CQRE [Secure] ’99*, volume 1740 of *Lecture Notes in Computer Science*, pages 43–63. Springer-Verlag, 1999.
15. Markus Jakobsson and Moti Yung. Revokable and Versatile Electronic Money (Extended Abstract). In *CCS ’96: Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pages 76–87, New York, 1996. ACM Press.
16. Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-Resistant Electronic Elections. In *WPES’05: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, pages 61–70, New York, 2005. ACM.
17. Dennis Kügler and Holger Vogt. Off-line Payments with Auditable Tracing. In *Financial Cryptography. 6th International Conference, FC 2002. Revised Papers*, volume 2357 of *Lecture Notes in Computer Science*, Berlin, 2002. Springer-Verlag.
18. Byoungcheon Lee, Colin Boyd, Kwangjo Kim, Jeongmo Yang, and Seungjae Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *In Proc. of Information Security and Cryptology (ICISC’03)*, volume 2971 of *Lecture Notes in Computer Science*, pages 245–258. Springer-Verlag, 2004.
19. Tatsuki Okamoto. Receipt-Free Electronic Voting Schemes for Large Scale Elections. In B. Christianson *et al.*, editor, *Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 25–35. Springer-Verlag, 1997.
20. Ronald Rivest and Warren Smith. Three Voting Protocols: ThreeBallot, VAV, and Twin. In *Proceedings of Electronic Voting Technology Workshop, 2007*, pages 1–14, Boston, MA, 2007.
21. Peter Y.A. Ryan. Prêt à Voter With a Human-Readable, Paper Audit Trail. Technical Report CS-TR: 1038, Newcastle University, 2007.
22. Peter Y.A. Ryan and S.A. Schneider. Prêt à Voter with re-encryption mixes. In *Computer Security—ESORICS 2006. Proceedings*, volume 4189 of *Lecture Notes in Computer Science*, pages 313–26, Hamburg, 2006. Springer-Verlag.
23. Shahrokh Saeednia, Steve Kremer, and Olivier Markowitch. An Efficient Strong Designated Verifier Signature Scheme. *Information Security and Cryptology — ICISC*, 2971:40–54, 2003.
24. Kazue Sako and Joe Kilian. Receipt-Free Mix-Type Voting Scheme: A practical solution to the implementation of a voting booth. In *Advances in Cryptology - EUROCRYPT’95. Proceedings*, pages 393–403, Berlin, 1995. Springer-Verlag.
25. Matt Smart and Eike Ritter. Remote Electronic Voting with Revocable Anonymity. Technical Report CSR-09-06, School of Computer Science, University of Birmingham, 2009. URL: <ftp://ftp.cs.bham.ac.uk/pub/tech-reports/2009/CSR-09-06.pdf>.
26. John Stewart. A Banana Republic? The Investigation into Electoral Fraud by the Birmingham Election Court. *Parliamentary Affairs*, 59(4):654–667, 2006.
27. Melanie Volkamer and Robert Krimmer. Secrecy forever? Analysis of Anonymity in Internet-based Voting Protocols. In *Proceedings, First International Conference on Availability, Reliability and Security, ARES 2006*, pages 340–347, Vienna, 2006. IEEE.
28. Stefan G. Weber, Roberto Araújo, and Johannes Buchmann. On Coercion-Resistant Electronic Elections with Linear Work. In *Proceedings, 2007 2nd Inter-*

national Conference on Availability, Reliability and Security, pages 908–16, Vienna, 2007. IEEE.