Prêt à Voter with Confirmation Codes

Peter Y A Ryan,
Interdisciplinary Centre for Security and Trust
and
Dept. Computer Science and Communications
University of Luxembourg
peter.ryan@uni.lu

Abstract

A scheme is presented in which a Pretty Good Democracy style acknowledgement code mechanism is incorporated into Prêt à Voter. The idea is to provide voters with an immediate, and usable, confirmation at the time of casting of the correct registration of their receipt on the Web Bulletin Board. As with PGD, the registration and revelation of the confirmation code is performed by a threshold set of Trustees. Verification of the registration of the vote is now part of the vote casting and therefore more immediate and convenient for the voters.

The scheme presented here is thus more convenient while maintaining the level of verifiability of conventional Prêt à Voter. It also means that we are less reliant on the diligence of voters in later performing checks on the Bulletin Board. It seems probable that this confirmation code mechanism will provide voters with greater confidence that their vote will be accurately tallied.

1 Introduction

Verifiable voting schemes seek to ensure that, not only are all legitimately cast votes correctly counted in the final tally, but that this be demonstrable to all participants and observers. Typically this is broken down into the verifiability of three steps:

1. cast as intended
2. recorded as cast
3. counted as recorded

In this paper we are primarily concerned with the second step: verifying that (encrypted) ballots are accurately registered and included in the tabulation. For this step, most verifiable voting schemes, Prêt à Voter [Rya05, CRS05] included, require voters, some time after casting their vote, to check on a public Bulletin Board that their encrypted ballot is correctly registered. Concerns are often raised about this: voters are expected to take extra steps beyond and after the casting of their vote and it is not clear that a sufficient number of voters will make the effort. Exactly what would constitute a "sufficient" number is itself not immediately clear: presumably it means: enough for there to be a significant chance of cheating to be caught and so deter cheating. How adversaries will perceive such risks is difficult to quantify.

Various mechanism have been proposed to try to counter this concern: provision of *voter helper organisations* to perform the checks on behalf of the voters, establishing a Verifiable Encrypted Paper Audit Trail (VEPAT), [Rya06], to allow auditors to check consistency between the audit trial and the WBB etc. In this paper we explore an alternative and more convenient mechanism to verify the correct registration of the ballot on the WBB.

This paper proposes the incorporation of a ballot registration confirmation mechanism into Prêt à Voter: at the time of casting her ballot the voter gets an immediate, human checkable confirmation, in the form of a confirmation code, that her receipt has been correctly registered on the Web Bulletin Board. As with Pretty Good Democracy, [RT08], the construction is such that receipt of the correct code by the voter is proof that the ballot has been correctly and jointly registered by a threshold set of trustees, subject to the assumption that codes have not been leaked. The voting procedure thus becomes a simple *vote, check and go.* Voters still have the opportunity later to visit the WBB to confirm that their receipt appears correctly, but the assurance of the integrity of the election is not so reliant on voter diligence in performing these checks.

For the first and last of the verification steps mentioned above, we use conventional techniques to achieve verifiability. The standard random audits of ballot forms serve to detect any ill-formed ballots that would lead to incorrect capture of voter intent. Once the election has closed and the set of registered, encrypted ballots has been assembled on the WBB, standard, universally verifiable tabulation techniques are used to ensure that all registered receipts are correctly decrypted and counted.

The fact that the scheme presented here provides two independent ways to verify the correct registration of ballots on the WBB, suggests that this scheme is strictly more secure than conventional Prêt à Voter. It might be argued that the confirmation code mechanism will give voters a, possibly false, sense of security and so might make them even less inclined to visit the WBB later than with conventional Prêt à Voter. The level of assurance of integrity provided by the confirmation code mechanism is not as high as that provided by the conventional checking of a receipt against the WBB, due the need to place a degree of trust in the Trustees. Thus, if the confirmation code mechanism were to be subverted, but leakage of the codes or a corrupt threshold set of Trustees, we might have lowered the overall assurance of integrity.

We argue that, suitably integrated in Prêt à Voter, this mechanism increases both the trustworthiness of the scheme and will help imbue a higher level of trust in the voters and other stakeholders. Note also that the presence of codes makes the checking of the receipts on the WBB less error prone: it will be harder to miss an incorrect code than a slightly shifted X.

We note that the implementation of the confirmation code mechanism proposed here is intimately linked to the implementation of the secure Bulletin Board concept. In both we anticipate using a distributed implementation involving a set of Trustees to update and sign the contents of the WBB.

The structure of the paper is as follows: after describing related work, the contribution and the claims and assumptions, we outline Prêt à Voter and PGD. We then describe the introduction of the PGD style confirmation code mechanism in Prêt à Voter. We discuss the possibility of doing away with the conventional receipt and WBB check mechanism and replacing this with a pure confirmation code mechanism. We then present some possible extensions: to deal with ranked voting and distributed printing of the ballot forms. After an analysis of the properties of the scheme we conclude.

2 Related Work

The scheme presented here is based on ideas from Prêt à Voter, [Rya05, CRS05] and Pretty Good Democracy, [RT08]. To our knowledge, the idea of incorporating a mechanism to provide immediate confirmation of correct registration has not previously been proposed for a polling station scheme, but it is of course quite common in remote schemes. The present scheme has some similarity to VoteBox, [SDW08], in that this also a verifiable scheme with a simple vote-and-go ceremony. VoteBox does not however provide the voter with a confirmation or a receipt but rather provides guarantees of correct registration via a mechanism of broadcasting ballots around a LAN and beyond along with a form of parallel testing.

The scheme presented here has some similarities to Scantegrity, [ea08], in that that also involves codes against each candidate. In both schemes these codes are initially kept secret, though here the reason is different: to ensure that the system cannot simply reflect the code back to the voter. As we discuss later, we can however also use the codes in a Scantegrity fashion for verification as long as we ensure, as with Scantegrity, that the voter only gets to learn the single, selected code.

In an accompanying paper in this volume "Acknowledgement Codes" by C. Culane *et at* builds on the scheme presented here and extends it to deal with ranked voting, Single Transferable Vote (STV) etc.

3 Contributions

We propose the introduction to Prêt à Voter of a mechanism that provides immediate confirmation that the receipt has been correctly registered on the WBB. The construction is similar to that employed in the internet voting scheme Pretty Good Democracy, [RT08], but the construction presented here allows us to handle full permutations of the candidates. Another innovation presented here is a construction that allows the distributed printing of the confirmation codes on the ballots. This serves to mitigate the threat of leakage of codes during the ballot printing process.

In Pretty Good Democracy, due to the unsupervised context, it was necessary to use a single ack code per code sheet in order to ensure receipt-freeness. The downside of this is that the verifiability is conditioned on the voting and acknowledgement codes not leaking. Now, due to the use of Prêt à Voter-style ballots, we can revert to using distinct codes for each candidate without violating receipt-freeness. As a result, this scheme, like Prêt à Voter, provides full E2E verifiability: even an adversary with knowledge of confirmation codes cannot alter votes in an way that will be undetectable.

4 Outline of the Claims and Assumptions

Here we state the key claims for the scheme along with the principle assumptions. We will present a more detailed analysis of the scheme with respect to various threat models in section 13.

4.1 Assumptions

We will make the usual assumptions for such polling station schemes: an accurate electoral roll is maintained and suitable mechanisms are in place to authenticate legitimate voters and prevent multiple voting. We assume the existence of a secure Bulletin Board: which allows append-only and which guarantees a consistent view of its contents to anyone who queries it.

For the confirmation code mechanism the scheme's security guarantees (detailed in the claims below) rely on a subset of the following trust assumptions holding (in addition to the standard assumption that the cryptography is secure):

1. No set of colluding, corrupt trustees is a "threshold set".
2. The confirmation codes are not revealed to any of the trustees before the election (this includes an assumption about the security of the printing process).
3. Only a single confirmation code is ever revealed (to voters and trustees) for voted ballots.

The third assumption is only necessary if we incorporate the Scantegrity style of checking and challenging the registration of votes.

We require algorithms admitting re-encryption and, for definiteness, we will assume that this is ElGamal. Given that votes are cast by flagging pre-committed material material on the WBB, we will ignore related plaintext style attacks on the privacy of mixes [PP89]; the adversary cannot cast re-encryptions or other transforms of previously cast ballots.

We will also set aside threats of ballot stuffing for the purposes of this paper and in effect trust that polling station will not send signed ballots to the Trustees that do not correspond to ballots cast be legitimate voters. A simple measure here is to post the names of voters recorded as having cast a vote on the WBB, but without any association between names and ballots. Voters whose name appears but who did not vote can challenge and anyone can check that the number of ballots matches that of the registered voters.

4.2 Claims

The scheme inherits the properties of Prêt à Voter: individual verifiability of the ballot construction, universal verifiability of the tabulation and ballot secrecy. More precisely:

1. (Integrity) If the election terminates successfully (and no complaints were recognized as valid), the announced tally matches the intended votes of all honest voters. This holds regardless of the trust assumptions.
2. (Augmented Integrity or "Vote and Go Integrity"): If Assumptions (1) and (2) hold, then integrity is assured even if voters do not check their receipts on the WBB.
3. (Ballot Secrecy): If Assumption (1) holds, then no adversary can guess the a voter's choice (better than an ideal adversary who is given the final tally and votes of corrupt voters).

The correctness of the ballot construction, i.e. that the voter's choice is correctly encoded, is ensured in the usual fashion by auditing by independent observers of randomly selected ballots before, during and after the election.

Correctness of registration of votes is assured by the voters visiting the WBB after casting their vote to (re-)confirm that their ballot appears correctly, and challenging using their receipt if it does not appear correctly. In this scheme we have an additional mechanism that serves to confirm correct registration of the ballots: the confirmation codes. Under the additional assumptions mentioned above, the only way that the correct code can be returned to the voter is if the ballot is correctly registered on the WBB by a threshold set of Trustees.

Note that, even if additional assumptions are compromised, so undermining the confirmation code mechanism, we still have the back-up of checking the WBB later. Thus, suppose that, prior to casting, the polling station discovers the confirmation codes for a ballot form, it could alter the vote and still announce the correct code to the voter. However, in this case the information posted to the WBB would not be consistent with the receipt, and this would be detectable in the usual Prêt à Voter fashion.

Finally we need to be sure that all registered ballots are correctly decrypted and included in the final tally. For this we use the standard verifiable anonymising mixes and threshold decryption mechanisms, [Nef01, JJR02, HS00]

Besides the verifiable integrity properties the scheme also guarantees the secrecy of all ballots, receipt-freeness and coercion-resistance. These are largely as for Prêt à Voter, but we need to check that the additional mechanisms introduced here do not undermine the properties. Subject to the stated assumptions, the presence of an additional random code alongside the X on the receipt and on the WBB provides a coercer with no additional information beyond what would be available to him anyway in Prêt à Voter. Whereas leakage of the codes could undermine the confirmation code mechanism, it will not compromise ballot secrecy. This depends on maintaining the secrecy of the candidate order for each ballot in the usual Prêt à Voter fashion. Assuming that this information is not leaked, the cryptography is secure and the link to the left hand portion of the ballot forms is lost then the adversary cannot determine how a voter cast her vote.

Obelix	
Idefix	
Asterix	
Panoramix	
	7304944

Figure 1: Prêt à Voter ballot form

X
7304944

Figure 2: Prêt à Voter ballot receipt (encoding a vote for "Idefix")

5 Outline of Prêt à Voter

The key innovation of the Prêt à Voter approach is to encode the vote using ballots with a randomised candidate list. Suppose that our voter is called Anne. At the polling station, Anne chooses at random a ballot form sealed in an envelope; an example of such a form is shown in Figure 1.

In the booth, Anne extracts her ballot form from the envelope and makes her selection in the usual way by placing a mark, e.g. a X in the right hand column against the candidate of her choice (or, in the case of a Single Transferable Vote (STV) system for example, she marks her ranking against the candidates). Once her selection has been made, she separates the left and right hand strips along a thoughtfully provided perforation and discards the left hand strip. She is left with the right hand strip which now constitutes her *encrypted ballot*, as shown in Figure 2.

Anne now exits the booth clutching her encrypted ballot, registers with an official, and casts her receipt. Her receipt is placed over an optical reader or similar device that records the serial number at the bottom of the strip and records in which cell her X is marked. The scanner produces a counterfeit proof, digitally signed receipt which is franked and returned to her to keep. The original is cast in a ballot box as a backup. Additional copies can be made available to observers and auditors. The code in the scanner/printer should be kept to a minimum to facilitate verification, but it will be advised that voters and observes check that the copies are accurate representations of the information on the original receipt: the serial number and position of the X.

The randomisation of the candidate list on each ballot form ensures that the receipt does not reveal the way she voted. Incidentally, it also removes any bias towards the candidate at the top of the list that can occur with a fixed ordering.

The value printed on the bottom of the receipt, that we refer to as the ballot serial number, is a hash of a ciphertext whose plaintext defines the candidate order printed on the ballot. The ciphertext along with the hash value is committed to the WBB during the setup phase. (The ciphertext could also be printed on the ballot.) The encryption is performed under a public key with the secret keys threshold shared across a number of (Decryption) Tellers. Thus, only a threshold set of these Tellers acting together are able to interpret the vote encoded on the receipt.

After the election, voters (and perhaps proxies acting on their behalf) can visit the secure Web Bulletin Board (WBB) and confirm their receipts appear correctly. Once any discrepancies are resolved, the Tellers take over and perform anonymising mixes and decryption of the receipts. All the intermediate stages of this process are committed to the WBB for later audit. Various auditing mechanisms are in place to ensure that all the steps, the creation of the ballot forms, the mixing and decryption etc are performed correctly. These are carefully designed so as not to impinge on ballot privacy. Full details can be found in, for example, [CRS05, Rya08]

A useful feature of Prêt à Voter is that the voter does not need to communicate her choice of candidate to any device. This is in contrast to most other verifiable schemes and has the advantage of neatly sidestepping any threats of the device leaking this information via some side-channel.

6 Outline of Pretty Good Democracy

Pretty Good Democracy (PGD), [RT08], is an enhancement of Chaum's Code Voting, [Cha01], that, subject to the assumption that codes do not leak, renders code voting end-to-end verifiable. The key idea is to threshold secret share the knowledge of the codes amongst a set of Trustees. Each voter is provided, via an assumed secure channel such as the post, with a *code sheet.* Each code sheet has the list of candidates and for each candidate it has a random vote code. Each code sheet also carries a single acknowledgement code. To vote, the voter logs onto a voting portal and provides the serial number of her code sheet along with the vote code for her chosen candidate. A threshold set of Trustees cooperate to confirm that the code is valid, register it on the WBB and then reveal the acknowledgement code.

Prior to the start of the election, a table is committed to the WBB. Each row of this table corresponds to a code sheet and is indexed by the serial number on the code sheet. The entries of the table are pairs comprising an ElGamal encryptions under the threshold public key of the Trustees PK_T of: 1/ a candidate index, 2/ a vote code. Each such pair corresponds to a row of the code sheet with the matching candidate, vote code. However, the order in which these triples appear in the WBB table are secretly permuted with respect to the (standard) order on the code sheets. An additional column on the table contains the encryptions of the acknowledgement codes.

When the voter sends in her serial number i and vote code VC, the server encrypts the code under PK_T and posts this $EVC := \{VC\}_{PK_T}$, along with a Zero Knowledge proof of knowldege of the plaintext, alongside the ith row of the table. The Trustees now check the ZK proof and perform Plaintext Equivalence Tests of EVC against the encrypted vote code terms in the row. If they find a match this confirms that the code is valid and the corresponding cell is flagged. The acknowledgement code is threshold decrypted and returned to the server to be relayed to the voter. The ZK proof is to avoid an attack in which a re-encryption of a posted term is submitted.

Once the election is over, tabulation can proceed: for all flagged cells in the table the encrypted candidate index term is extracted and entered into the anonyising mix.

7 Introducing Confirmation Codes into Prêt à Voter

The following players are involved in the unfolding of the election:

Voters: those eligible to cast votes.

Election Authorities: tasked with defining the election parameters: candidates, electoral roll and etc.

Mix Tellers: who perform the anonymising (re-encryption) mixes of the encrypted candidate indices.

Decryption Tellers: who decrypt the mixed, encrypted candidate indices.

Trustees: who are involved in the registering of votes and the revealing of the confirmation codes.

Auditors: Assumed independent of the above and drawn from various factions: the political parties, the Electoral Commission, Election Observers etc. They are tasked with performing various auditing functions.

7.1 The Initial Setup of the Election

The cryptographic setup we use here is similar to that of Pretty Good Democracy (PGD), [RT08], in that we commit a table to the Web Bulletin Board that is used to register votes as they are cast. Integrity of the election depends on ensuring the consistency of the information committed to the WBB with what is printed on the ballots. We start by describing the table that is committed to the WBB before the election starts, and then go into the details of the distributed constructions that lead to this.

At the start of the election we have a table, that we refer to as the S table, posted to the WBB. Suppose that we have n candidates and ν voters and λ is a multiplier to allow for random auditing. If we expect to audit up to half the ballots for example then $\lambda = 2$. The S table will have n columns (ignoring the indexing column) and $\lambda \cdot \nu$ rows.

Each row corresponds to a ballot form, is indexed by the serial number of the ballot, and comprises a secret permutation ρ_i of the candidate indices encrypted under the Teller's public key PK_{Te} paired with the confirmation codes, CC, encrypted under the Trustee's public key PK_{Tr}. The candidates are represented by indices 1 to n. Thus, the i-th row of the table has the form:

$$i, (\{\rho_i(1)\}_{PK_{Te}}, \{CC_{i,1}\}_{PK_{Tr}}), (\{\rho_i(2)\}_{PK_{Te}}, \{CC_{i,2}\}_{PK_{Tr}}), \dots, (\{\rho_i(n)\}_{PK_{Te}}\{CC_{i,n}\}_{PK_{Tr}})$$

The ballot forms are similar to the usual Prêt à Voter ballots except that now they have three columns: there is an addtional column to the right that carries the confirmation codes covered by scratch strips. The order of the indices and the confirmation codes in the i-th row must match the candidate order printed on the i-th ballot form. Similarly, the confirmation codes printed down the right hand column must match the sequence of codes encrypted across the ith row. Thus, for $n = 5$, the ith ballot form will have the form, ignoring the scratch strips for the moment:

Candidate	Vote "X"	Conf Code
$Candidate_{\rho_i(1)}$		$CC_{i,1}$
$Candidate_{\rho_i(2)}$		$CC_{i,2}$
$Candidate_{\rho_i(3)}$		$CC_{i,3}$
$Candidate_{\rho_i(4)}$		$CC_{i,4}$
$Candidate_{\rho_i(5)}$		$CC_{i,5}$
		i

Figure 3: A blank ballot (scratch strips not shown)

We now describe the steps required to reach this goal in a distributed fashion.

7.2 The Distributed Construction

Firstly the Election Authorities construct the P table. This will have n columns (ignoring the indexing column), where n is the number of candidates and $\lambda \cdot \nu$ rows. Each row of the P table comprises encryptions under the threshold public key of the Decryption Tellers, PK_{Te}, of the candidate indices $\{1, ..., n\}$ in numerical order. The P table is posted to the WBB. Thus the ith row has the form:

$$i, \{1\}_{PK_{Te}}, \{2\}_{PK_{Te}}, \dots, \{n\}_{PK_{Te}}$$

Note that the construction of the P table can be public and verifiable. We could for example perform the encryptions using randomisation $r = 1$. Now each row is subjected to a sequence of independent, verifiable, secret re-encryption shuffles performed by the Mix Tellers to yield the Q table:

$$i, \{\rho_i(1)\}'_{PK_{Te}}, \{\rho_i(2)\}'_{PK_{Te}}, \dots, \{\rho_i(n)\}'_{PK_{Te}}$$

Where $\{M\}'$ denotes a re-encryption of $\{M\}$.

For some secret permutation ρ_i. Note that each row of the Q table is the result of the application of a sequence of secret permutations applied by each the Mix Tellers and so no strict subset of these will know the final permutation.

7.3 The Confirmation Codes

We now form the R table that carries the encrypted confirmation codes. We employ a distributed construction similar to that used in Pretty Good Democracy. Suppose that we are using four digit confirmation codes. The Election Authorities start by generating a suitable number of such codes, $\lambda \cdot n \cdot \nu$ to be precise, and encrypt these under the Trustee's public key PK_{Tr}. These are then subjected to a number of re-encryption shuffles by the Mix Tellers. The resulting shuffled encrypted codes are now assembled into a table with n columns and $\lambda \cdot \nu$ rows, indexed with the serial numbers.

Finally we form the S table as the join of the Q and R tables. So now the (i, j) cell of the S table contains:

$$(\{\rho_i(j)\}_{PK_{T_e}}, \{CC_{i,j}\}_{PK_{T_r}})$$

The result is the S table described above. All the steps leading from the P table to the R table are posted to the WBB for subsequent auditing.

8 Printing the Ballot Forms

We need to extract the candidate orders and confirmation codes from the rows of the S table and print these to the ballot forms to be distributed to the voters. Unfortunately, the creation and distribution of the ballots results in a vulnerable step that could lead to the leakage of information regarding the candidate orders and codes. The pleasing distributed construction of this information on the WBB is rather undermined by the need to get this information to the voters in an easily usable form. For the moment we will simply assume trusted processes to perform the decryption, printing and distribution, i.e we assume these do not to leak any information. Later we will discuss ways to distribute the process and so spread the trust and weaken this assumption.

Firstly, for the ith ballot, we decrypt the codes and print these down the right hand column in the order that they appear across the ith row of the S table. This requires a threshold set of Trustees to decrypt the $\{CC_{i,j}\}_{PK_{T_r}}$ terms. Once printed, each of these codes are covered by a scratch strip.

Now we need to print to the candidates down the left hand strip in the order encoded in the ith row of the R table. One approach is to do this in an *on-demand* fashion, along the lines described in [Rya08]. We will not pursue this here but assume that this is done in advance. For this we require a threshold set of the Decryption Tellers to be available to extract the candidates.

In summary, the ballot bearing the serial number i will now have the candidates printed down the left hand column in the order corresponding the permutation encoded in the i-th row of the WBB. The centre column will be blank, ready for use by the voter. In the right hand column, the j row will carry the (i, j)-th confirmation code $CC_{i,j}$, as indicated in figure 3. However, the codes will not in fact be visible but will be covered by scratch strips as indicated in figure 4. We will describe approaches to distributing these processes in section 15.1.

9 The Voting Ceremony

We confine our discussion for the moment to the situation in which voters simply choose a single candidate. The voting ceremony is initially as for conventional Prêt à Voter:

- The voter, let's call her Anne, enters the polling station and pre-registers: she presents identification, is confirmed as a legitimate voter and is given a ballot form at random sealed in an envelope. This is noted in the register against her name. She is cordially reminded to leave the scratch strips intact.
- Anne goes to the booth, extracts the ballot from the envelope and puts her X in the middle column against her candidate of choice and detaches and discards the left hand column.
- She exits the booth and goes back to the voting desk where, in the presence of officials and observers, her ballot is scanned and the serial number and index of the marked cell it is digitally signed and sent to the Trustees.
- Her vote is registered by a threshold set of Trustees and the confirmation code (along with the serial number and index value) are signed and returned to the polling station. This information printed out and handed to the voter.
- The scratch strip is removed from the right hand cell next to the X and the confirmation code revealed. If this matches the code just returned

by the Trustees the vote is considered to have been correctly registered.

Note that the casting of the vote and the checking of the returned confirmation code can be performed in the presence and with the assistance of officials and independent observers. If there is a discrepancy they are on hand to witness this, report it and take appropriate action.

Assuming all goes smoothly, i.e. the correct confirmation code is returned, the printout with the serial number, index and confirmation code is given to the voter to retain. The original ballot is dropped in a ballot box as a back-up.

Candidate	Vote "X"	Conf Code
Geoffroy		■■■■
Nicolas		■■■■
Rufus		■■■■
Alceste		■■■■
Clotaire		■■■■
		35899325

Figure 4: A blank ballot form

Candidate	Vote "X"	Conf Code
Geoffroy		■■■■
Nicolas	X	■■■■
Rufus		■■■■
Alceste		■■■■
Clotaire		■■■■
		35899325

Figure 5: The voted ballot

If, as discussed later, we use a Scantegrity style challenge mechanism, it is important that to ensure that only the one code is revealed, all the other codes should remain concealed under intact scratch strips. The copy of the receipt handed to the voter should thus carry no information about the other codes. This is to provide additional protection to the voter later in the event of a challenge, in the manner of Scantegrity II, [ea08].

Vote "X"	Conf Code
	■■■■
X	■■■■
	■■■■
	■■■■
	■■■■
	35899325

Figure 6: The receipt prior to scanning

Vote "X"	Conf Code
	■■■■
X	4909
	■■■■
	■■■■
	■■■■
	35899325

Figure 7: The receipt with the confirmation code revealed

9.1 Vote Registration

On scanning a receipt, the polling station posts to the WBB alongside the ith row of the S table, a signed term with the serial number and the index indicating the marked cell: $Sig_{PS}(i, j)$.

A threshold set of the Trustees check signature and, assuming that it is valid, flag j-th cell of the ith row of the S table. They then threshold decrypt the $\{CC_{i,j}\}_{PK_{Tr}}$ term and return the following term to the polling station: $Sig_{Tr}(i, j, CC_{i,j})$. The Polling Station will check the signature and print out a form with the serial number, the index and the returned confirmation code.

9.2 Tabulation

Tabulation is much as for PGD: once the election has closed and any challenges resolved, all flagged encrypted candidate terms in the S table are collected together and entered into a standard, verifiable anonymising mix followed by threshold decryption. The collecting of the flagged terms is universally verifiable and so, subject to the usual assumptions about the WBB, we are guaranteed that exactly the registered ballots will be accurately included in the final tally.

10 Error Detection and Recovery Mechanisms

If there are malfunctions or corruption, the voting protocol may not run exactly according to the intended sequence described above. Here we discuss some of the key failure modes.

The first problem that might arise is that the confirmation code does not arrive within a prescribed time limit. This may indicate problems with the availability of a threshold set of Trustees or communication channels. In this case we will have to resort to fall-back procedures, for example recording the ballot locally and trying to register again later. It may be possible to arrange for the confirmation code to be later emailed or texted to the voter.

The confirmation code returned does not agree with the code revealed on the ballot form. The first step here is to visit the WBB and check what exactly has been registered. If the correct cell has been flagged and the correct code revealed then at least the vote has been correctly registered and the error must lie in the return step of the protocol.

If the wrong cell in the S table has been filled in, then the term posted to the WBB by the Polling Station should be checked. If the signature is correct but the wrong index or even wrong serial number is included then the blame can be laid on the Polling Station.

If there is a failure of the assumption underpinning the confirmation code mechanism, it may be that the correct code is returned and yet the wrong cell in the S table is flagged, or nothing is registered. This will be detectable, either by the voters themselves or observers doing spot checks using either the paper audit trail or additional copies of the receipts. If this occurs then it must be immediately investigated and it may be that the confirmation code mechanism is suspended at this polling station.

Due to space limitation we do not go into an exhaustive discussion of the failure modes and recovery strategies here. This will be dealt with in a follow-on paper devoted to the robustness of the scheme.

11 Auditing

As with all E2E verifiable schemes, for the assurance of accuracy we want to avoid as far as possible the need to place any trust in the components that execute the voting process. Thus, we ensure that any error or corruption that could affect the outcome will be detectable. Furthermore, we need to ensure that the cause of the error can be diagnosed and the culprit identified. In this section we outline the auditing procedures designed to detect and diagnose such errors.

The P table can be constructed in a publicly verifiable fashion, as mentioned earlier. The row shuffles applied to the P table to obtain the Q table should be verified as genuine shuffles, to ensure that each row contains a (secret, encrypted) permutation of the indices 1 through n. The construction of the R table from the Q table is a simple, universally verifiable step. In fact, such audits may not be strictly necessary as the random audits described below would detect ill-formed rows in the S table.

For the integrity property of the scheme it is essential to ensure that the ballot forms are consistent with the S table. That is to say: for a ballot with serial number i, the candidate order printed on the ballot should match the sequence of the encrypted candidate index terms in the i row of the S table. Also the sequence of confirmation codes printed down the right hand column should match the sequence of (encrypted) confirmation code terms in the i-th row of the table.

This is probably best achieved in the usual fashion by preprinting an excess number of ballots and having independent auditing entities randomly selecting a proportion and checking these for consistency with the corresponding row of the S table. Such random audits would be performed before, during and after the election.

For a ballot selected for audit the serial number would be submitted to the Trustees with a clear indication that this is for audit purposes. Such a request would be signed by the auditor and the signature verified by the Trustees. The Trustees would also verify that this serial number has not previously been used to cast a vote. A threshold set of the Trustees would then decrypt the codes publicly on the WBB. In similar fashion, the Tellers would decrypt the candidate terms. The auditors could them verify consistency between the what is printed on the ballot form and what is revealed on the WBB.

Voters could be offered the opportunity to audit a ballot handed to them rather than use it to vote. The ballot serial number could be visible through a window in the envelope. For an audited ballot the serial number would be sent to the WBB and a threshold set of trustees would reveal the candidate order and confirmation codes and send this back to the polling station. The ballot would then be extracted from the envelope and the lists compared.

Clearly care is needed to enforce mutual exclusion of voted and audited ballots. For example, for audited ballots the message to the WBB could include an "audit" string rather than the usual index value used for a voted ballot. Another possibility is to use the two sided ballot proposal of [Rya07]: each ballot form has an independent Prêt à Voter ballot on each side. We will not go into the details here.

Anti-counterfeiting and chain of custody measures would be employed to make it hard to alter ballots or inject fake ballots.

The actions of the Tellers, identifying flagged terms and including them in the mix, mixing and decrypting these terms, can all be auditing in the standard fashion. That all flagged terms are included in the first column of the mixes is straightforwardly and universally verifiable. The correctness of the mixes can be verified using standard techniques, e.g. partial random checking, [JJR02], or Zero Knowledge style, e.g. [Nef01]. Finally the correctness of the decryptions can be verified using ZK proofs, e.g. [CP93, PBD07]

12 The Role of Receipts

The mechanisms used to authenticate a receipt have always been delicate with voter-verified schemes. Suppose that the polling station is required to apply a digital signature to each receipt to authenticate it as corresponding to a validly cast ballot. A corrupt Polling Station device might apply a false signature on a receipt which could result in a voter's legitimate challenge being called into question. Typically it is suggested that voter helper organizations or similar would provide a signature checking service at the polling station, but this might always be practical to ensure.

It is natural to ask if the confirmation code mechanism could completely replace the receipts as a way to obtain assurance of correct registration. This has a number of arguments in its favour: firstly we avoid the issue of false challenges being raised based on faked receipts or genuine challenges being cast into doubt by a false digital signature. Secondly, we avoid the concern that voters might not believe the privacy of their vote is assured by the encryption of the receipt. The latter might be quite a reasonable concern given that most current encryption algorithms might be broken in 10-20 years say.

The downside of doing away with the receipt and WBB checking mechanism is that we would be totally reliant on the confirmation code mechanism. Given that this is dependent on the assumption that the codes do not leak and the absence of threshold collusion of corrupt Trustees this seems very dangerous.

We could do away with the digital signatures on cast ballots and rely on non-cryptographic anti-counterfeiting measures: special paper and printing along with franking of cast ballots. Such mechanisms have the advantage that they are immediately verifiable by humans.

We note that there are alternatives to making counterfeit proof receipts: firstly, a mechanism along the lines of the Verified Encrypted Paper Audit Trail of [Rya06]. Such an audit trail could be made secure by, for example printing it to a till roll format and perhaps using a chained hash techniques to make tampering with the record difficult. We could also use a VoteBox like mechanism to broadcast ballots to a LAN and beyond.

Secondly, we could use the challenge mechanism of Scantegrity II [ea08]: the voter only gets to learn the confirmation code corresponding to their candidate of choice, i.e. we require that all other confirmation codes remain concealed by the scratch strips. Now, if the voter finds that the code on the WBB differs from the one she has noted she can protest and, as with Scantegrity II, there is a procedure to open the commitments to all the confirmation codes for that ballot. If the code the voter is claiming is indeed a valid code then there are strong grounds for believing that the the complaint is genuine. If the code claimed by the voter is not a valid code then it is likely that the voter is either mistaken or trying to discredit the election. Thus we are less reliant on making the receipts hard to forge. As with Scantegrity, this mechanism could be undermined if voters are able to get get hold of alternate codes in some other way.

13 Security properties of the Scheme

The scheme presented here is essentially Prêt à Voter-with an additional mechanism to confirm correct registration of encrypted ballots. This suggests that the scheme is at least as secure as Prêt à Voter. We should note however that the confirmation code mechanism might result in fewer voters making the effort to verify their ballot on the WBB. Given that the level of assurance provided by the confirmation code is lower than that provided by the WBB checks, this could result in an effective overall lowering of the level of security. Note however that all errors or corruption remain detectable, the only issue is whether the probability of detection has been lowered. It would be wise therefore to maintain separate audit trails of cast ballots that auditors can check against the WBB record. This might be a VEPAT style mechanism as mentioned earlier and could include the generation of extra copies of the ballots at the time of casting that are made available to observers.

The auditing mechanisms for the ballot construction and tabulation phases are essentially identical to those of conventional Prêt à Voter.

As far as ballot secrecy is concerned, the addition of a random code alongside the voter's X on the receipt and the WBB gives an adversary no additional information. On the other hand, the presence of the code adds to the integrity properties: it is now harder to alter any flagged cells on the S table as this would have to be accompanied by the creation of a valid ZK proof of decryption of the corresponding code.

The back-end construction is rather different from the original versions of Prêt à Voter. In the latter, the accuracy property follows purely from the correct construction of the ballots. In the scheme described here, we must ensure consistency between what is committed to the WBB and what is printed on the ballots. Arguably for both, any corruption would be detected by random audits, but it is clear that the checks are rather more delicate for the current scheme: we need to ensure that the auditor does get an accurate representation of the WBB for example. The secure broadcast property of the WBB should ensure this of course.

14 Possible Enhancements

Here we describe some possible enhancements to the scheme.

14.1 Distributed Printing of the Ballot Forms

In this section we propose a construction that allows distributed printing of the confirmation codes, i.e in such a way as to ensure that no single entity knows any code in its entirety. There are many ways that one might imagine distributing this process, for example using distributed printing with invisible ink in the manner of [ECHA09]. Alternatively one could consider using multiple channels with some procedure for combining shares at the time of vote casting. The first is technologically complex and the latter increases the complexity of the voting ceremony. The approach proposed here seeks a pragmatic balance between security on the one hand and simplicity on the other. Thus the codes are not genuinely secret shared but rather each Clerk knows a fragment (digit).

Suppose for the purposes of illustration that we use three digit codes, accordingly we nominate three *Code Clerks*, call them S_1, S_2, S_3, with public keys PK_{S_1}, PK_{S_2} and PK_{S_3}. S_1 will be responsible for decrypting and printing the first digits of each the confirmation code, S_2 the second digit and so on.

The construction is similar to that presented earlier except that now, rather than just encrypting the codes we also create encryptions of each digit under the public keys of the Code Clerks. We start as we did for the generation of the R table: we generate a sufficient number of three digit codes and again form the encryption of each code under the public key of the Trustees. In addition we encrypt each digit under the public key of the appropriate Code Clerk. Suppose that confirmation code CC has the digits: CC_3, CC_2 and CC_1. We now form the tuple:

$$(\{CC\}_{PK_{Tr}}, \{CC3\}_{PK_{S_3}}, \{CC2\}_{PK_{S_2}}, \{CC1\}_{PK_{S_1}})$$

These tuples are now put through as series of mixes as before but now the tuples are preserved by the shuffles. After shuffling, these tuples are assembled into the R^* table. Thus the i, jth cell of the R^* table will contain the following terms:

$$(\{\rho_i(j)\}_{PK_{Te}}, \{CC_{3,i,j}\}_{PK_{S_3}}, \dots, \\ \{CC_{1,i,j}\}_{PK_{S_1}}, \{CC_{i,j}\}_{Tr})$$

S_1 will start the ballot printing process: for the ith ballot it will take the (i, j)-th cell of the R^* table, and will decrypt the 1st digit of the confirmation code: $CC_{1,i,j}$. This digit will be printed to the jth row of the ith ballot form and then covered with a scratch strip. Similarly for for all the columns of the ith rows. S_1 will repeat this for all the ballots. S_2 will now take over repeat this process for the 2nd digits of the confirmation codes, and similarly for S_3.

The result of this process is that the confirmation codes are printed onto the ballot in such a way that any Code Clerk knows at most one of the digits of any given code. A possible attack is that S_2 removes the strips applied by S_1, notes the value and re-applies a scratch strip. Anti-counterfeiting measures could be used to counter this, for example, using differently coloured strips and distinctive patters for each Code Clerk. Alternatively we could use the invisible ink technology from Scantegrity II for which it would be hard to undo the revealing process. The Clerks could of course reveal the previous codes and then fresh ballots with the codes concealed again. To counter the threat might be to use special stock paper with anti-counterfeiting measures that would not be available to the Clerks. None of these measures seems entirely satisfactory and alternative ways to distribute the printing process is the topic of future research.

Printing of the candidates to the ballot forms requires a threshold set of Tellers to decrypt the $\{\rho_i(j)\}_{PK_{Te}}$ terms. One way to spread the trust is to rotate the Teller who performs the last step of the decryption. This will ensure that no single Teller sees all the permutations.

Another possibility is to introduce another authority, let's call him the *notary* N, whose role is to mask the link to the ballot serial numbers. N will cover each serial number i with a scratch strip and overprint an independent serial number i'. It retains a record of the association between the serial numbers and keeps this secret. Now when the Tellers are about to print the candidates to ballot with the visible serial number s' they pass this to N who then returns a re-encryption of the $\{\rho\}$ terms in the row indexed with s.

14.2 Handling STV and Ranked voting

The mechanisms described above do not adapt immediately to other voting methods such as ranked, STV etc. The problem is that a corrupt Polling Station or man-in-the-middle could reorder the ranking sent to the Trustees and then apply the inverse re-ordering to the returned codes. A simple way of countering this is to use confirmation codes of variable, randomly chosen lengths. Suppose that codes can have length 2, 3 or 4, chosen at random.

In the example above, suppose that the voter enters the ranking:

Candidate	Vote "X"	Conf Code
Geoffroy	5	■■■■
Nicolas	1	■■■■
Rufus	2	■■■■
Alceste	4	■■■■
Clotaire	3	■■■■
		35899325

Figure 8: A ranked ballot

The ranking vector $(5, 1, 2, 4, 3)$ is transmitted to the WBB. The Trustees open the confirmation codes and return the concatenated string: 493343940887534. The Trustees do not reveal how this breaks down into the individual codes until later. It is now very difficult for any adversary between the voter to reorder the ranking undetected. Now the voter reveals the codes on the ballot and checks that the string returned is correct. Note again that officials and observers can help the voter with this check.

An alternative way to deal ranked voting is presented in the accompanying paper in this volume "Acknowledgment Codes" by C Culane et al.

Candidate	Vote "X"	Conf Code
Geoffroy	5	34
Nicolas	1	4933
Rufus	2	439
Alceste	4	75
Clotaire	3	4088
		35899325

Figure 9: A ranked ballot with confirmation codes revealed

14.3 Countering Randomisation Attacks

In its simplest form, Prêt à Voter is vulnerable to randomisation attacks: the attacker demands that the voter come out of the polling station with a receipt with the X in the first cell for example. Counter-measures have been proposed, for example giving voters access to more than one ballot (which might be part of a voter ballot auditing process anyway), but these complicate the voting ceremony. It is not clear in practice how serious such attack would be, but we note that a variant of this scheme is resistant to randomisation, which we outline here:

As in PGD, an extra permutation is introduced in the S table, so that now the order of the candidate indices does not match that on the ballot form. We also use voting codes rather than simply relaying the cell index to the board. As with PGD, the trustees perform PET tests against the committed codes, flag the cell that matches and decrypt the corresponding confirmation code. The receipt given to the voter now comprises just the serial number and the confirmation code. Given that the voter has no control over or prior sight of the code, he cannot be expected to comply with any randomising demands from the coercer.

15 Conclusions

We have presented an elaboration of Prêt à Voter that is more convenient than conventional Prêt à Voter:

the basic ceremony is now a simple *vote, check and go.* We have argued that this scheme is at least as secure as Prêt à Voter and it seems probable that the confirmation code mechanism will add to voter confidence. Such a mechanism could be incorporated in other polling station verifiable schemes, for example Scantegrity II. If introduced with care, in particular not abondoning the usual WBB checks, this scheme gives strictly greater assurance of accuracy that conventional Prêt à Voter. We have also argued that the introduction of the confirmation code mechanism does not have any adverse impact on the ballot secrecy properties of Prêt à Voter.

The main arguments in favour of such a mechanism are: firstly the greater convenience for the voters, simple vote, check and go ceremony. Secondly, it seems probable that the confirmation code mechanism will also help with issues of stakeholder confidence. Thirdly, the presence of the codes should make the process of querying the WBB less error-prone: now the the voter can check not only the correct position of the X but also check the presence of the correct code. The additional redundancy in the code will make it less likely that a voter will miss an incorrect code than say a slightly shifted X.

A potential downside is the need for a threshold set of trustees to be available throughout the voting period. We note however, that it seems likely that we will in any case require a similar mechanism to implement the secure WBB, i.e. have a threshold set of Trustees or similar to be available on-line throughout the voting period. In this paper we have not gone into the details of exactly how the Trustees cooperate to register votes or how the secure WBB is implemented. It is clear that the implemention of the confirmation code mechanism and of the secure WBB are be closely related. This will be investigated in future work.

We have touched on the issues of error handling, accountability and recovery strategies, but clearly a much more exhaustive study needs to be performed to ensure that the scheme is sufficiently robust.

We have discussed the pros and cons of dropping the receipt and WBB checking mechanisms in favour of a pure confirmation code mechanism but have concluded that this would be unwise and risky. It would be interesting however to explore the possibility of not providing voter with a conventional, hard to forge, receipt and rely instead on VEPAT style challenges perhaps along with a Scantegrity style challenge mechanism in the hands of the voters.

16 Acknowledgements

The author would like to thank Dalia Khader, Gabriele Lenzini,Steve Schneider and Vanessa Teague for many insightful comments. Thanks also to Chris Culnane for suggesting extending the scheme to ranked voting.

The work reported here was funded by the Luxembourg Fonds National de Recherche (FNR) under the SeRTVS project.

References

[Cha01] D. Chaum. SureVote: Technical Overview. Proceedings of the Workshop on Trustworthy Elections (WOTE '01), 2001.

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 89–105, London, UK, 1993. Springer-Verlag.

[CRS05] D. Chaum, P.Y.A. Ryan, and S. Schneider. A practical, voter-verifiable election scheme. In *European Symposium on Research in Computer Security*, number 3679 in Lecture Notes in Computer Science. Springer-Verlag, 2005.

[ea08] D. Chaum et al. Scantegrity II: End-to-End Verifiability for Optical Scan Elec-

tion Systems using Invisible Ink Confirmation Codes. Technical Report CS-TR-1107, University of Newcastle upon Tyne, 2008.

[ECHA09] Aleks Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. How to print a secret. In *Proceedings of the 4th USENIX conference on Hot topics in security*, HotSec'09, pages 3–3, Berkeley, CA, USA, 2009. USENIX Association.

[HS00] Martin Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In *Advances in Cryptology*, number 1807 in Lecture Notes in Computer Science, pages 539–556. Springer-Verlag, 2000.

[JJR02] M. Jakobsson, A. Juels, and Ronald Rivest. Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking. In *USENIX Security Symposium*, pages 339–353, 2002.

[Nef01] A. Neff. A verifiable secret shuffle and its application to e-voting. In *Conference on Computer and Communications Security*, pages 116–125. ACM, 2001.

[PBD07] Kun Peng, Colin Boyd, and Ed Dawson. Batch zero-knowledge proof and verification and its applications. *ACM Trans. Inf. Syst. Secur.*, 10, May 2007.

[PP89] Birgit Pfitzmann and Andreas Pfitzmann. How to break the direct rsa-implementation of mixes. In *EUROCRYPT*, pages 373–381, 1989.

[RT08] P. Y. A. Ryan and V. Teague. Pretty Good Deomcracy. Proceedings of the Seventeenth International Workshop on Security Protocols 2009, to appear in LNCS, 2008.

[Rya05] P.Y.A. Ryan. A variant of the chaum voting scheme. In *Proceedings of the Workshop on Issues in the Theory of Security*, pages 81–88. ACM, 2005.

[Rya06] P.Y.A. Ryan. Verified encrypted paper audit trails. Technical Report Newcastle Tech Report 966, June 2006, University of Newcastle upon Tyne, 2006.

[Rya07] P.Y.A. Ryan. The computer ate my vote. Technical Report CS-TR-988, University of Newcastle upon Tyne, 2007.

[Rya08] Peter Y. A. Ryan. Prêt à Voter with paillier encryption. *Mathematical and Computer Modelling*, 48(9-10):1646–1662, November 2008.

[SDW08] Daniel R. Sandler, Kyle Derr, and Dan S. Wallach. VoteBox: A tamper-evident, verifiable electronic voting system. In *Proceedings of the 17th USENIX Security Symposium (USENIX Security 2008)*, pages 349–364, San Jose, CA, 2008.