vVote: A Verifiable Voting System

4 authors, including:

Chris Culnane
University of Melbourne
24 PUBLICATIONS   86 CITATIONS

SEE PROFILE

Peter Y. A. Ryan
University of Luxembourg
153 PUBLICATIONS   2,377 CITATIONS

SEE PROFILE

Steve Schneider
University of Surrey
170 PUBLICATIONS   3,983 CITATIONS

SEE PROFILE

vVote: a Verifiable Voting System

Version 2.0

Tuesday 30th September, 2014

Chris Culnane, Peter Y A Ryan,
Steve Schneider and Vanessa Teague

Contents

Abstract

The Prêt à Voter cryptographic voting system was designed to be flexible and to offer voters a familiar and easy voting experience. In this paper we present a case study of our efforts to adapt Prêt à Voter to the idiosyncrasies of elections in the Australian state of Victoria. This technical report includes general background, user experience and details of the cryptographic protocols and human processes. We explain the problems, present solutions, then analyse their security properties and explain how they tie in to other design decisions. We hope this will be an interesting case study on the application of end-to-end verifiable voting protocols to real elections.

This is the 17th September 2014 version of the "Draft Technical Report for VEC vVote System", incorporating additional material following the external review of the VEC vVote system.

The team involved in developing the vVote design described in this report were: Craig Burton, Chris Culnane, James Heather, Rui Joaquim, Peter Y. A. Ryan, Steve Schneider and Vanessa Teague.

1. Introduction

The potential advantages of electronic voting are obvious but the risks are not. The challenge is to obtain the obvious advantages for convenience and accessibility while preserving reasonable vote privacy and producing a rigorous evidence trail that will stand up to dispute.

This technical report details a proposal for end-to-end verifiable voting in Victoria, based on the Prêt à Voter end-to-end verifiable voting system [RBH+09].

1.1. End-to-end verifiability

End-to-end verifiability consists of three pieces of evidence:

Cast-as-intended verification: Each voter gets evidence that their vote is cast as they intended,

Counted-as-cast verification: Each voter gets evidence that their vote is included unaltered in the tally,

Universally verifiable tallying: Everyone can check that the list of (encrypted) cast votes produces the announced election outcome.

This project will not achieve verifiability all the way to the announcement of the election result, because it will run alongside an existing paper-based system that does not provide a universally verifiable tallying proof. See 1.5 for more details. In summary, the vVote system provides

- cast-as-intended verification,

- counted-as-cast verification and

- an output list of votes, with a universally verifiable proof.

Other forms of electronic voting (or indeed of paper-based voting) might provide some, but not all, of these properties. For example, some Australian states offer a simple electronic ballot marker, in which the voter uses a computer to help them fill in, and then print, a ballot that is put into a ballot box and counted along with the ballots that are completed manually. This provides cast-as-intended verification for sighted voters, because the voter can look at their printed ballot and check that it reflects their intention. It does not provide counted-as-cast verification, instead relying on procedures

5

for the secure transport of paper ballots (like the paper system does). It delegates tallying verification to scrutineers who observe the paper count (if there is one).

The VEC opted for Prêt à Voter over the simpler alternative of a plain electronic ballot marker because it provides for electronic transfer of ballot information from distant supervised locations, supported by verifiable evidence of correctness. This was regarded as particularly important for distant polling places (*e.g.* overseas) and for allowing any voter to vote at any polling place. Since this project commenced, a problem in the transport of West Australian Senate ballot papers in the 2013 federal election has focused national attention on the security of processes for transporting paper ballots.

1.2. Prêt à Voter overview

Prêt à Voter uses a ballot form that is printed before voting, with a list of candidates printed in a random order, and an encrypted version of the same list. Voters select or number the candidates by filling in boxes adjacent to the candidate names (in the Victorian protocol, they will have computerised assistance). They keep the list of marked boxes and the encrypted candidate list, and shred the human-readable candidate list. The two main properties of Prêt à Voter are privacy and end-to-end verifiability.

End-to-end verifiability is achieved in vVote, following the Prêt à Voter structure, as follows:

Cast-as-intended verification:
- **ballot printing confirmation** Each voter has the opportunity to confirm that the printed candidate lists on some ballots match the encrypted version on the same ballot.
- **preference printing confirmation** Each voter writes their own preferences (or in the case of vVote, checks that their own preferences are correctly written) on the other side of the ballot.

Counted-as-cast verification: Each voter gets the opportunity to check that their (encrypted) ballot appears in a public list of received votes,

Universally verifiable tallying: Everyone can check that the list of (encrypted) cast votes produces the announced output votes, by downloading and checking a public electronic proof.

1.3. Necessary procedures

Academic papers can simply state the assumptions under which the protocol is secure; practical deployments must ensure that those assumptions are met, or the system may not protect privacy or provide evidence of a correct election result. This technical report details the procedures that need to be followed in order for the system to attain the verifiability and privacy properties necessary for state elections.

End-to-end verification depends on voters performing proper procedures for verifying that their vote is cast as they intended and included in the count. All voters must be explicitly encouraged to perform some ballot confirmations on the printed ballots before voting, to check that the computer-assisted preference list matches their intended vote, and to look up their vote later on the public list of accepted votes. If the announced election outcome is disputed, then the evidence supporting it depends crucially on the number and distribution of these confirming procedures that were actually performed. For example, if there is one polling place in which voters were not informed of the ballot confirmation procedure, then all of the votes cast at that polling place lack any supporting evidence.

Privacy depends on the immediate destruction of the human-readable candidate list. Unlike ordinary completed ballots, the human-readable candidate lists also include a unique serial number that matches the one on the voter's receipt. Hence the candidate lists prove how individuals voted. Shredding the candidate list must be enforced at all polling places, immediately after the person votes.

1.4. The protocol

End-to-end verifiable election protocols are well studied in the academic literature, but until recently have not been deployed in public elections. In 2011 the Victorian Electoral Commission (VEC) approached the Prêt à Voter team to investigate adapting the scheme to the special requirements of the Victorian parliamentary elections, which use both Instant Runoff Voting (IRV, also called "alternative vote" in the UK, and simply "preferential voting" in Australia) and 5-seat Single Transferable Vote (STV)[1]. The first version of the final system has been recently delivered to the VEC at the time of writing, and a process of testing and review will soon be underway. This document is intended as an aid for testing, review and security analysis. The system is expected to be ready for the next Victorian State election in November 2014.

The proposed protocol is universally verifiable, meaning that there are no trust assumptions for guaranteeing the integrity of the votes.[2] There are probabilistic assumptions about the number of voters who confirm correct printing of some Prêt à Voter ballots, the number who check that their printout matches their intended vote, and the number who check that their receipt appears on the Web Bulletin Board (WBB). It also provides voters with evidence of malfeasance, assuming that they check the signature on their receipt before they leave the polling station.

Since this is a polling-station scheme, we do not address eligibility verifiability. Prevention of ballot stuffing is by existing procedural mechanisms.

The main departure from standard Prêt à Voter is the use of a computer to assist the user in completing the ballot. This is referred to as an "electronic ballot marker" (EBM). The EBM is trusted for vote privacy, which is different from standard Prêt à Voter in which the voter does not need to communicate her vote to any (encryption) device. This

---

[1]For more information on various election methods, please refer to the appendix in [XCH+10].

[2]Vision impaired voters must assume that at least one device reads accurately to them.

modification is necessary for usability, because a vote can consist of a permuted list of about 30 candidates. It seemed infeasible for a voter to fill in a Prêt à Voter ballot form without assistance. Indeed, simply filling in an ordinary paper ballot with about 30 preferences is a difficult task.[3] Computerised assistance is an important benefit of the project, and trusting the device for privacy seemed an almost unavoidable result of that usability advantage.[4] Hence our scheme depends on stronger privacy assumptions than standard Prêt à Voter. Providing privacy for complex ballots is notoriously difficult, and is further complicated by some details of Victorian elections that are described below. Our system provides privacy and receipt-freeness under reasonable assumptions about the correct randomised generation and careful deletion of secret data, and of course assuming a threshold of decryption key sharers do not collude. It does not fully defend against the "Italian Attack," or all other subtle coercion issues, but neither does the current paper-based system. We make this more precise below.

1.5. Challenges of Victorian Voting

Prêt à Voter was designed originally for first-past-the post voting, in which each voter chose a single candidate [CRS05]. Subsequent papers extended the scheme to more complex types of elections [RS06, Rya08, RBH+09, XCH+10].

The state of Victoria, like many other Australian states, runs simultaneous elections for two houses of parliament, the Legislative Assembly (LA) and the Legislative Council (LC), both of which use ranked-choice voting. Each LA representative is elected by IRV with compulsory complete preference listing, with rarely more than 10 candidates. Members of the Legislative Council (LC) are elected in 5-member electorates using STV. Voters typically choose from among about 30 candidates—they rank at least 5, and up to all candidates in their order of preference.

Because LC voting is quite complex, voters are offered a shorthand called "Above the line" (ATL) voting, which allows them to select a complete preference ranking chosen by their favourite political group (usually a party). Each political group (of which there are about 12) registers a (complete) STV vote in advance with the electoral commission. When someone votes ATL and chooses that group's ticket, this is equivalent to copying out their STV vote.

Traditionally, both LC voting options are presented on the same ballot paper. The ATL group selections are presented on top of a thick line (hence the name); the full STV options are shown below the line (and hence called "below the line" (BTL) votes).

Each polling place must accept votes from a resident of anywhere in the state. Hence

---

[3]Since some people deliberately vote informally, it is difficult to say exactly what percentage of people accidentally disenfranchise themselves by incorrectly filling in their vote. About 2% of votes in the 2006 state election were ruled informal because of "numbering errors" [Vic07] , but the overall informality rate is closer to 10%, especially when there are lots of candidates on the ballot. See https://www.vec.vic.gov.au/Results/stateby2012distributionMelbourneDistrict.html for an example.

[4]In principle one could use an EBM to fill in a series of ballots and only cast one of them, without telling the device which one. This is too much work for voters.

our system must produce Prêt à Voter ballots for every electoral division in both the LA and the LC, available at every polling place. This is a significant challenge for Prêt à Voter, but Prêt à Voter confers the great advantage of verifiability on these votes. The existing methods of verifiable paper counting do not work with this requirement. For the large fraction of people who vote outside their home electorate, completed paper ballots must be sent to the home electorate by courier, usually arriving after the polling-station count has been completed and after observers have departed.

This system will not be responsible for all of the votes cast in the upcoming state election, so it will have to combine with existing procedures for casting and counting ordinary paper ballots. For LA and LC-ATL votes this is straightforward. However, LC-BTL votes are complicated. Even those cast on paper must be tallied electronically—in the existing system they are manually entered and then electronically tallied. The authorities then make complete vote data available to allow observers to check the count.[5] This is why the system does not achieve a complete universally verifiable tally. The proof that this system produces would be sufficient for end-to-end verifiability if it carried all votes in the election, but it is not possible to do STV tallying (whether verifiable or not) on a subset of votes. As it is, the scrutineers who observe the paper count will have to check that the publicly verifiable output from vVote matches the votes that are added to the paper count.

Preferential elections are vulnerable to coercion through signature attacks [DC07], commonly referred to as Italian attacks. The system proposed here does not address this attack, primarily because it will work alongside a paper system that is also susceptible to it. Our system also reveals whether a person voted ATL or BTL. This is unlikely to have political consequences.

Another challenge is producing an accessible solution for voters who cannot fill out a paper ballot unassisted. This is a primary justification for the project, but producing a truly verifiable solution for such voters is extremely difficult, because many of them cannot perform the crucial check that the printout matches their intention (though see [CHPV09] for a verifiable and accessible protocol). We provide a way for them to use any other machine in the polling place to do the check, in which case the cast-as-intended property depends upon at least one of the machines in the polling station not colluding with the others to manipulate the vote.

1.6. Specific design choices

1.6.1. Cast-as-intended verification

Wombat [BNFL+12], VoteBox [SDW08] and several other polling-station end-to-end verifiable voting schemes guarantee integrity by using "Benaloh challenges," [Ben06] which require filling in the vote more than once. This would be time-consuming for 30-candidate STV. It would perhaps be possible to make challenges easier (for example, by letting the device remember the last vote), but the integrity guarantees still depend on the voter per-

---

[5]These procedures are also under review and improvement, but are out of the scope of this paper.

forming quite a subtle randomised protocol. We have opted for Prêt à Voter, in which voters may confirm the correctness of the unvoted ballot form. This process (called "auditing" in older versions of Prêt à Voter) is independent of the vote size and can be completed with assistance without compromising privacy. It also provides full accountability: there is no need to take the voter's word for how they voted. A ballot confirmation check that completes with an invalid proof can be used as evidence; an attempted ballot confirmation check that does not complete at all can have multiple (human) witnesses.

1.6.2. Unified Scanner and EBM

We have already described why completing the ballot needs to be assisted by a computer. Our original design [BCH+12a] included separate steps for filling in the ballot and then scanning the printed receipt. This was designed to separate the information of how the person voted from the knowledge of what their receipt looked like: the EBM learnt how the person voted, but could not subsequently recognise their ballot (and hence link it to the individual voter), while the scanner knew the receipt but did not know the corresponding plaintext. However, user studies at the VEC determined that a three-step voting process was too cumbersome for use. Also the necessity of print-on-demand meant that there was already an Internet-connected machine in the polling place that was trusted for maintaining privacy of the information on the printed ballot, including which candidate ordering corresponded to which receipt. For both these reasons, the new protocol now unifies the job of the scanner and the EBM, though it retains a separate print-on-demand step. The voter first collects their ballot form, and has an opportunity to perform a confirmation check on it, then goes to an EBM to fill in the ballot, then the EBM sends the receipt electronically and also prints a paper record for the voter to check. This now means there are two online manchines in the polling place (the EBMs and the ballot printers) that are trusted for vote privacy.

1.6.3. Print on Demand

This project necessitated a new protocol for the verifiable printing on demand of Prêt à Voter ballot forms. The crucial requirement is that voters (and others) can perform a confirmation check on some ballot forms for correctness *without compromising voter privacy*, because the check occurs before the person votes. Voters then vote on ballot forms that have not had a confirmation check. Ballot forms that have had a confirmation check cannot be voted on because their associated ciphertexts have now been decrypted, and privacy would be lost on such votes.

The integrity of Prêt à Voter depends crucially on proper construction of the printed ballot forms, meaning that the plaintext candidate list that the voter sees must match the encrypted values for that ballot. This technical report details opportunities for confirming their correct construction and printing. The construction is very computationally efficient and retains most of the desirable properties of existing print-on-demand proposals in the literature. The information flow of our scheme is similar to Markpledge 2 [AN06], though the confirming is different. The main idea is that the device encrypts the vote

directly using randomness generated by others. The protocol was first presented in JETS [CHJ$^+$13].

Also the system must address the question of "kleptographic" privacy attacks [GKK$^+$06], in which the (public) ciphertexts contain deliberately poorly-chosen randomness that exposes the vote. This is possible whenever the entity building the ciphertexts also controls all the randomness used. This problem is addressed by distributing the process of inputting randomness into ciphertexts.

This proposal is designed so that the entity that builds the ciphertexts (the printer) has a deterministic algorithm to follow. This provides a way to distribute the expensive cryptographic operations to the network of printers, whilst retaining the central, distributed, generation of randomness to maintain the following three properties:

- ensuring the candidate lists are randomly generated,

- ensuring no single generating entity knows all the (plaintext) candidate lists, and

- ensuring extra information about the candidate list cannot be leaked in the ballot ciphertexts (as in kleptographic attacks).

We then devise a confirming mechanism to ensure correctness of the printed ballot forms.

1.6.4. Randomised Partial Checking

The exact choice of mixnet is independent of other aspects of the protocol, but in this implementation we have selected a mixnet based on Randomised Partial Checking [JJR02]. This was partly due to efficiency, and partly to the ease of explaining to the public how the mixnet works.

However, improvements in both the implementation and the efficiency of zero-knowledge shuffling proofs [TW10, FS01, Nef01] could make them a reasonable alternative in future versions. In theory they have superior properties, because their privacy and soundness are stronger, can be proven formally, and rely on weaker assumptions than those of RPC. However, they remain computationally intensive and difficult to explain.

1.7. Related Work

In the USA, voter verifiable paper trails with auditing (VVPAT) are a common means of achieving software independence. However, this does not solve the problem of secure custody and transport of the paper trail. Furthermore, performing rigorous risk-limiting audits seems intractable for IRV [MRSW11, Car11], let alone for 30-candidate STV.

The most closely related project is the groundbreaking use of Scantegrity II in binding local government elections in Takoma Park, MD [CCC$^+$10]. Our project has very similar privacy and verifiability. However, both the number of votes and the complexity of each ballot are greater for our system. Although the Scantegrity II scheme appears to have been highly successful in the context of the Takoma Park elections, Prêt à Voter

is more appropriate for our application. Scantegrity II is inherently for single-candidate selections. It has been adapted to IRV in Takoma Park by running a separate single-candidate election for each preference, but would be difficult to adapt to 30-candidate preference lists. Even with computer assistance, a 30 by 30 grid of invisible ink bubbles seems too complicated for most voters.

The StarVote project in Texas [BBB+13] represents an interesting combination of end-to-end verification techniques and risk limiting audits [Sta09].

1.8. Prior work and paper overview

In a previous paper [BCH+12a] we gave an overview of this project, including the context of Victorian voting and some ideas about how we would implement the protocol. A followup version [BCH+12b] gave more details and some preliminary security analysis. The print on demand protocol was presented in [CHJ+13]. Here we present all of the protocol, including both the cryptographic protocol and the human procedures to be followed in the polling place and at the electoral commission. Our aim is for a comprehensive analysis of the protocol's security, including the assumptions on which privacy depends, a precise explanation of the kind of verifiability achieved, and a clear statement of the issues that remain.

An overview including the main system components and an account of the voting experience is contained in Section 2, which includes a statement of the main security claims. Many of the system's security properties depend on proper procedures in the polling place—these are detailed in Section 3. The individual system components are described in Section 4. Section 5 contains mechanisms for achieving robustness in the presence of certain failures. A comprehensive and rigorous threat analysis is contained in Section 6.

2. System Overview

2.1. System Components

The system has the following main components:

Public Web Bulletin Board (Public WBB): an authenticated public broadcast channel with memory.

Private Web Bulletin Board (Private WBB): a robust secure database which receives messages, performs basic validity checks, and returns a signature. Validly signed messages are guaranteed, under certain assumptions, to appear subsequently on the Public WBB.

Print-on-demand printer: a combination of a computer and printer which generates Prêt à Voter ballots in advance of the election, then prints them on demand.

Randomness Generation Service: a collection of servers that produce randomness for the print on demand process.

Electronic Ballot Marker (EBM): a computer that assists the user in filling in a Prêt à Voter ballot.

Cancel Station: a supervised interface for cancelling a vote that has not been properly submitted or has not received a valid Private WBB signature.

Mixnet: a set of (preferably independently managed and hosted) Mix servers that produces a noninteractive, universally verifiable proof of a shuffle and decryption (of encrypted votes) and posts it to the Public WBB.

Election key sharers: authorities who share the key used to decrypt votes at the end of the election.

2.2. Voter Experience Overview

This section gives an overview of the verification process, listing all the steps for a voter to verify that their vote is cast as they intended and properly included in the count, and also the process for universal verification of the output.

Recall that voters cast an IRV vote for a Legislative Assembly district and then for their Legislative Council region either a full STV vote or an 'ATL' shorthand. (See Section 1.5). A printed ballot therefore consists of:

- A human-readable serial number (shortened to SerialNumber below),

- a human-readable district name (which also determines the region),

- a human-readable randomly ordered list of the candidate names for the LA district,

- a human-readable randomly ordered list of the candidate names for the LC region,

- a human-readable list of group names (for LC-ATL voting),

- a QR code containing all this data, plus a WBB digital signature on it.

Three races: a named district race and the two exclusively votable parts of the region race (a group-list vote or voting for individual candidates) A list of candidate names for the District and candidate voting part of the region race A list of group names for the group-list part of the region race

Figure 2.1 shows the ballot form on the left side, and also the preferences as printed out by the EBM. The process for receiving the ballot form, then casting a vote on it while verifying that the vote matches the voter's intention, is as follows:

Pollworker: authenticates the voters (using whatever method is traditional) and sends a print request to the Print On Demand device specifying the district/region they can vote in,[1]

Printer: retrieves and prints appropriate ballot, including SerialNumber and district, with private WBB signature.[2]

Voter (Check 1): may choose to check and confirm this ballot. This involves demanding a proof that the ballot is properly formed, *i.e.* that the permuted candidate list corresponds correctly to the ciphertexts on the public WBB for that serial number. If the ballot has a confirmation check, the voter returns to the printing step for a new ballot.

Voter: shows the printed ballot barcode to the EBM, then enters the vote via user-friendly EBM interface,

EBM: prints on a separate sheet:

   1. the electoral district,

   2. the SerialNumber,

   3. the voter preferences permuted appropriately to match Prêt à Voter ballot,

   4. a QR code with this data, plus private WBB signature.

   This is the voter's receipt. An example is shown on the right side of Figure 2.1.

---

[1]More generally, it is the pollworkers' responsibility to authenticate the voter, request the ballot(s) that the person is eligible to vote on, and record how many people voted for each division.
[2]Details on what is signed and how are in Section 4.2

Voter (Check 2): checks printed preferences against printed candidate list, and checks that the district is correct and the SerialNumber matches that on the ballot form.

Voter (Check 3): (optionally) checks the WBB signature, which covers only data visible to the voter. This requires an electronic device.

If either of Check 2 or 3 fails, the vote is cancelled using the cancellation protocol of Section 3.1.4.

Voter: shreds the candidate list,

Voter: leaves the polling place.

Voter (Check 4): later checks their vote on the public WBB. They only need to check the serial number and order of their preference numbers.

Anyone (Check 5): after polling closes, checks the universally verifiable proof that all submitted votes are properly shuffled and decrypted.

The rest of this Technical Report expands on each of these steps so as to give a complete account of end to end verification and an analysis of privacy.

CANDIDATE LIST

THIS IS NOT A BALLOT

This list must not
leave the polling place.

<<

Serial Number PRA:538

Shepparton

(Not in ballot paper order)

| Tomato, Tessa
VEGETABLES PARTY |
| Hamburger, Harold
MINCEMEAT PARTY |
| Singapore, Style
NOODLES PARTY |
| Strawberry, Siva
ICECREAM PARTY |
| Blue, Belinda
CHEESE PARTY |
| Chardonnay, Claude
FINE WINES PARTY |
| Multigrain, Mario
BREAD PARTY |

Northern Victoria

(Not in ballot paper order)

GROUPS

| FINE WINES |
| CHEESE |
| GOOD COFFEE |
| ICECREAM |
| BREAD |
| NOODLES |
| VEGETABLES |
| MINCEMEAT |

(Not in ballot paper order)

CANDIDATES

| Brie, Bernadette
CHEESE PARTY |
| Turnip, Thomas
VEGETABLES PARTY |
| Sauvignon, Cabernet |

PREFERENCES RECEIPT

THIS IS NOT A BALLOT

VEC Voting receipt
My vote remains private

>>

Serial Number PRA:538

Shepparton

(Not in ballot paper order)

4	NOTE: These boxes will be empty if you didn't vote in the district ballot.
5	
3	
2	
7	
1	
6	

Northern Victoria

(Not in ballot paper order)

GROUPS

	NOTE: These boxes will be empty if you voted for individual candidates.
1	

(Not in ballot paper order)

CANDIDATES

	NOTE: These boxes will be empty if you voted for a party or group.

Figure 2.1.: Separate vote printouts (truncated at the bottom). The voter collects the
left side (the candidate list) when they are marked off. The EBM then prints
the right side with the voter's preferences. Note the matching serial numbers.

2.3. Security Properties

The intention is to provide a proof of integrity independent of any trust assumptions, while preserving reasonable privacy.

We have several different kinds of security assumptions, which apply at different points:

Computational assumption: A reduction to a computational problem generally believed to be hard. For example, the privacy of ElGamal encryption relies on the hardness of the Decision Diffie Helmann problem on the elliptic curve being used. The soundness of the zero knowledge proofs of correct decryption, which use the Fiat-Shamir heuristic, relies on the assumption that inverting the hash function is as hard as inverting a random oracle.

Auditing assumption: An assumption that a sufficiently large and unpredictable fraction of a set have been confirmed or checked. For example, proper ballot generation, proper ballot printing, and accurate printing of voter preferences all need to be checked with high enough probability and unpredictability to give us confidence in the accuracy of those that were not checked.

Threshold or distributed assumption: An assumption that a known threshold of authorities will not misbehave. For example, votes on the public WBB are private as long as fewer than a threshold of the authorities who share the decryption key collude, and not all the mixers collude. Robustness and reliability of the private WBB are also dependent on threshold assumptions. The design requires a threshold greater than 2/3 of the number of peers. For example, if there are 7 peers then a suitable threshold is 5.

Individual trust assumption: Trusting a single device. For example, the EBM a person uses to vote is trusted not to leak the vote. The printer is trusted not to leak ballot information.

Obviously, the intention is to minimise instances of trusting a single device. Our system design aims to provide

Integrity based only on computational or auditing assumptions. If a sufficient number of confirmation checks are properly conducted then the election's integrity is demonstrated given only computational assumptions.[3]

Non-repudiation based on a threshold assumption. Unless more than a threshold of private WBB peers collude, it should be infeasible to produce a properly signed receipt without properly casting a vote.

---

[3]The computational assumptions are due only to the use of the Fiat-Shamir heuristic to choose challenges for the proof of a shuffle. An alternative method of generating unpredictable challenges based on some other assumption could also be used, in which case there would be no computational assumptions for integrity.

Robustness based on a threshold assumption. If a threshold of private WBB peers behave properly, a properly signed receipt is guaranteed to appear on the public WBB.

Privacy is the most subtle property, and needs to be discussed separately at several points. We assume that the link between an individual and their receipt is public (though names are not printed on the WBB).

The printer is trusted not to leak ballot information via side channels.

The EBM is trusted not to leak the vote via side channels.

The printer is prevented from performing kleptographic attacks by the ballot generation confirmation check. The proper generation of randomness for those ballots depends on at least one of the randomness generation servers being honest.

The encrypted votes on the WBB remain private under threshold assumptions on the decryption key sharers and an assumption that there is at least one honest mix server.

The system does not as it stands defend against pattern-based coercion attacks (a.k.a. "Italian attacks"), or other subtle coercion techniques such as forced randomisation.

Verification of voter eligibility is dependent on human procedures: the system is dependent on a secure procedure for ensuring that only eligible voters can vote, with at most one vote each. We assume that at some point a ledger of how many people have voted in each division at each polling place is reconciled with the published list of encrypted votes on the WBB.

We emphasise that for verification purposes all the voters and other observers have access to the public WBB, which is broadcast on a reliable channel. Hence there are no trust assumptions for integrity (apart from voter eligibility and polling-place ballot stuffing).

There are, however, threshold trust assumptions for liveness, reliability, and non-repudiation. In other words, we rely on certain thresholds to prevent certain kinds of failures, although all those failures would be detectable even if all the authorites misbehaved. (Whether they would in practice be detected might depend on an auditing assumption.) The private WBB peers provide a robust database implementation that distributes trust so the electoral authorities will only publish something that is verifiable as long as the trust assumption holds. Since the authorities are responsible for choosing the peers to trust, they are responsible for meeting that assumption.

Another way of looking at it is that the voters themselves do not need to trust anything for integrity, because they can verify it. The authorities want to be confident that what they publish will indeed verify. The design tells the authorities that, under certain trust assumptions, the system will give them what they need, and hence satisfy the interested and sceptical members of the public who want to verify the outcome.

The confirmation checks involved in verifiability also provide a way of catching bugs or errors in the software: a failed check might also be due to a coding error, and successful checks also demonstrate the absence of coding errors that could affect the result of the election.

The protocol uses digital signatures to provide evidence of many kinds of failures (in addition to the more traditional end-to-end verifiability literature, which tends to focus on detection alone). This provides two kinds of benefits: voters can prove that a malfunction occurred, but not persuade anyone that a malfunction occurred when it did not. This is important in defending against the "defaming attack" in which people pretend to have detected a system failure which did not actually happen. Of course, there can be no proof that the EBM accurately represented the voter's intention: that step is dependent on the voter's testimony and hence is to an extent vulnerable to the "defaming" attack.

3. Procedural details

This section details, from the human perspective, how certain important security conditions are enforced by insisting on particular human procedures. (The next chapters explain how electronic processes guarantee other security properties.) The most important procedures in the polling place include giving each voter exactly one, correct, ballot, allowing them to chose some at random to perform a confirmation check on, encouraging them to check their printed vote and its signature, and insisting that they shred their candidate list. The procedures and guarantees for vision impaired voters are slightly different from those for sighted voters, because checking the printout requires the use of a device.

This chapter describes what checks should be performed to test for normal operation. Recovery from failures is described in Chapter 5.

3.1. Typical voters

3.1.1. Getting a ballot

The voter presents herself to an official at a polling station and her name is marked off a register. The official sends the print station a request for a ballot of the appropriate LA and LC division. The print station prints the ballot with a Private WBB signature. (Print station is abbreviated as VPS in other documentation.)

> **It is essential for privacy that no-one except the voter sees the association between the candidate order and serial number on the ballot, so printing should be private.**

Obviously it is essential for integrity that each voter is allowed to vote at most once, on a ballot of the appropriate division. This must be enforced by procedures at the polling place.

> **Reconciling the number of marked-off voters in each division with the number posted on the WBB is essential for preventing ballot stuffing.**

Check 1a: Confirming ballot correctness. Once she has obtained her ballot, the voter should decide whether she wishes to run a confirmation check on it or use it to vote. A confirmation check, called "auditing" in previous versions of Prêt à Voter, means checking that the encrypted list of candidates on the WBB matches the plaintext candidate ordering on the RHS of the ballot. Ballot confirmation ensures that the ballot is well-formed and hence would correctly encode a vote. We describe the ballot confirmation

procedure below in Section 3.1.3. She can repeat the ballot confirmation procedure as many times as she wants in principle, each time obtaining a fresh ballot, until proceeding to vote using the last obtained, unconfirmed ballot. This implements an iterated cut-and-choose protocol: not knowing which option the voter will choose before committing to the printed ballot serves to counter any attempts by the system to manipulate votes by issuing malformed ballots. Confirming ballot construction necessarily reveals encryption information, so a ballot that has been confirmed should not be subsequently re-used for voting.

> **It is essential for integrity that all voters have the opportunity to perform a confirmation check on as many ballots as they wish.**

The more voters who check, the stronger the evidence that the ballots are well formed.

Check 1b: Checking the WBB signature on the printed ballot. Each ballot is printed with a WBB signature to indicate it is legitimate. Voters should check this signature before voting—if they vote on an illegitimate ballot *i.e.* one which did not originate properly from the PoD protocol, their vote will not be accepted and their privacy could be breached. This does not in itself affect integrity, because an attempt to deprive a person of a vote by giving them an illegitimate ballot will be detected at voting time when the WBB refuses to sign their submitted vote.

3.1.2. Casting a vote

Assuming that she is now happy to proceed to casting her vote, the voter takes the last obtained ballot to the booth. In standard Prêt à Voter she would now proceed to fill in her preferences directly on the ballot. However, given that the LC-BTL section contains about 30 candidates, it is not reasonable to expect the voter to enter her ranking preferences using a permuted candidate list. Instead we propose to use a touch screen Electronic Ballot Marker (EBM) that will display the candidates in standard order, as previously introduced in [BCH⁺12a]. The voter enters her preferences via the screen in the standard way, then the EBM permutes them to match the candidate permutation on her ballot. This means that we have to sacrifice one of the pleasing features of standard Prêt à Voter: that no device directly learns the voter's choices. This seems unavoidable for such expressive ballots if the system is to be usable.

She inserts the ballot into the EBM and selects her preferred language and can run through a training module on the machine to learn about the whole voting procedure, verification and tallying. The voter is now offered the choice of sequence in which she votes that is, the Legislative Assembly (LA) or Legislative Council ballots, and for the latter she can vote either "above the line" (ATL) or "below the line" (BTL). Note that although the voter can vote at any polling station, the LA ballot is specific to the region in which she is registered. She must however, fill in both a LA and LC ballot and will be prompted by the EBM to ensure that she does this.[1]

---

[1]Exact rules on ballot spoiling are a matter of user interface: at present, voters are allowed to cast incomplete or invalid preference lists, as long as they are warned. The receipts then reveal their

For each ballot (LA or LC), the EBM scans the the QR code which represents the permutation of the candidate ordering on her ballot and displays the candidates in legal ballot order. Once the voter enters her choices, she is asked to confirm her choices and when she does so, the EBM prints on a separate sheet of paper:

1. the district,

2. the SerialNumber,

3. the voter preferences permuted appropriately to match the Prêt à Voter ballot,

4. a QR code with this data, plus private WBB signature.

This is the voter's receipt. Note that the EBM knows the permutation on the ballot and so re-orders the voter's selection accordingly. Note also that the EBM can assist the voter by pointing out syntactic errors, for example, duplicate rankings etc.

Before printing, the EBM submits to the Private WBB exactly the data it will print on the receipt. Then at printing time it adds the Private WBB signature, as a further QR code, onto the receipt.

Check 2: EBM vote printing. The voter should check that the printed receipt matches her intended vote. This includes checking that the serial numbers match, and that the printed preferences match her intended vote arranged according to the candidate order on her ballot.

> **It is essential for integrity that all voters are encouraged to check that their printed vote matches their intention.**

Once she has checked her printout, the voter shreds the candidate list. This prevents her from proving how she voted.

> **It is essential for privacy that all voters are required to shred their candidate list before leaving the polling place.**

Existing laws preventing voters from photographing their ordinary paper ballots should also apply to the candidate list, for the same reason: a voter who retains evidence of the order the candidates are listed on her ballot can prove later how she voted.

Check 3: Private WBB Signature on vote. The voter can check the signature using a purpose-built smart phone app. This must of course incorporate a check that the data signed by the WBB is the same as the data printed on the paper.

> **It is essential for non-repudiation that the voter checks the signature on her receipt before leaving the polling place.**

---

decision to spoil their ballot. An alternative, but not currently implemented, method, would be to include a candidate called "spoiled ballot" who would be the first preference of any invalid ballot. Subsequent preferences would be meaningless, but could be filled in to make the receipt look like that of a valid vote. This would hide whether the voter had voted formally or not.

If she fails to check, and does not receive a properly-signed receipt, then she will be able to detect later, but not to prove, that her properly-submitted vote has been excluded from the WBB.

It is strongly advisable to perform signature checking after the candidate list is shredded, so that voters do not have an opportunity to photograph their candidate list.

The voter should be easily able to produce multiple copies of her receipt, for example using a photocopier or a camera (on a smartphone). This combats the "trash attack," [BL11] and also allows others to check her receipt on the WBB. It would also be quite reasonable for the VEC to retain duplicate copies of receipts, as well as letting the voters take them home. Of course there would have to be a careful procedure for ensuring that the centrally retained reciepts were accurate copies of the voters'.

Check 4: Receipt appears on WBB. After a given time period, the voter can use her receipt to check that the information is correctly recorded on the WBB.

These 4 checks provide evidence that the vote is cast as the voter intended, and included unaltered in the count.

We now describe the ballot confirmation process in more detail.

3.1.3. Confirming ballot correctness

To perform Check 1, confirming ballot correctness, the ballot can be taken back to the printer. The printer prints a proof of correct ballot formation, along with a WBB signature. The WBB must record that the ballot has been confirmed, and therefore not accept any vote cast with that ballot form. As part of the confirmation process, a clear "CHECKED—NOT TO BE USED TO VOTE" message (which must be visible) is printed on the ballot form.

The voter can also check the proof of decryption later on any other machine, including at home, so we are not trusting the polling-place machines for confirmation of ballot construction.

When the day's WBB becomes available (see Section 4.1), it shows which serial numbers were confirmed and displays a proof of what the candidate ordering should be. (It also shows which ones were voted and what the preferences were.)

Ensuring the mutual exclusion of confirmed and cast ballots is vitally important. The Private WBB must run a realtime check that the same ballot is not both confirmed and voted. This process is trusted for privacy, but not for integrity because violations are detectable.

In Section 5 we describe what to do when some of these checks fail.

3.1.4. Ballot cancellation: individual quarantine

There may be legitimate circumstances when a voter finds a check is not successful and wishes to instruct that the vote should not be cast. This may be used for various failures, including

- when a voter claims the printed vote differs from their intention

- when a printed vote does not include a valid WBB signature

- when the EBM fails to produce a printout

- when an attempted ballot generation or ballot printing confirmation check fails, either because it times out or because it does not produce a valid proof of correctness or a valid signature.

The terminology used by VEC for this process is *Individual Quarantine (IQ)*. The Cancel Station posts a request that the vote should not be included in the count, and the vote is thus quarantined. Once the Cancel Station has been certified (at the end of the election) as behaving correctly and thus the cancellation requests are considered to be genuine, then the vote is removed. Malfunctoning or malicious cancel stations attempting to cancel votes invalidly are thus prevented from doing so. Vote cancellation requests are monitored to identify such behaviour, and any Cancel Station has a low limit on the number of cancel requests that can be made before manual intervention from VEC is required.

An IQ request overrides any other request, such as confirming or voting. When a vote is cancelled, the cancellation is recorded against the Serial number on the (private and public) WBB. The voter must give up their ballot receipt in order to request IQ, and is provided with an IQ receipt to show that the request will appear on the WBB. A voter's possession of a valid receipt is their evidence of the vote that they cast, and that they did not request to be withdrawn. Hence they must give up that receipt when requesting IQ.

> **A cancellation request must never be issued for a vote for which a voter has already left the polling place with a valid receipt.**

The intention and expectation is that this process is used rarely, and with the explicit observation by at least two (and preferably more) officials at the polling place. Cancellation requests should be independently recorded on paper at the polling place, and should require approval from senior officials. If the voter does not have a ballot receipt, for example because the EBM failed to produce a printout, then this will also be recorded.

3.1.5. Procedures for defeating chain voting

Chain voting, which applies to conventional voting too, is an attack in which a coercer smuggles a (partially) completed ballot out of a polling place, and then gives it to a voter with instructions to cast it and bring an unmarked ballot back out. The coercer then fills in this ballot and uses it to repeat the chain voting attack with a new voter.

In Prêt à Voter, the chain voting risk applies to a coercer who obtains a printed ballot form, records the candidate order, and then sends a voter into back into the polling place with instructions to vote in a particular way and return with both a receipt and a new, unmarked, ballot form. Since the coercer has already recorded the voter's candidate order, the receipt shows how the person voted. The new ballot form is used to repeat the attack with a new voter.

vVote includes some technical measures to defend against chain voting. Printed ballot forms expire after 5 minutes if they have not been used to start a session, and the private WBB refuses to allow the same ballot form to be used to start another voting session once it has been used to start one. This means someone who sneaks an unused printed ballot form out of the polling place has 5 minutes to send it in with another voter. If someone sneaks one out having used it to start a session (and the tablet sits there with session active), then attempting to sneak this back in will not work as the printed ballot cannot be used to start a fresh session and the abandoned session itself "locks.".

3.2. Vision impaired voters

We assume that the vision-impaired voter has registered at a polling place and had her name marked off. The printing station should work fine for vision impaired voters, though there may need to be special procedures to help them collect their ballot privately. If they need assistance, it is important that the assistant does not see the printed candidate list.

The vision impaired voter takes the slip to an Electronic Ballot Marker (EBM). At the EBM, she inserts the slip. The system is set so that she has an audio-only session in her preferred language and the touch screen is laid out like a keypad, following the TVS2 standard. For example, the four corners when touched render 1, 3, $*$ and $\#$, the middle top and bottom give 2 and 0, and so on.[2]

The session is similar to the one described previously in Section 3.1.2 in that the voter has to fill in ballots for her LA and LC (ATL/BTL) votes, but this time she indicates her choices by touching the appropriate parts of the screen and has voice prompts to guide her. When she has filled in all required parts of the slip, she is given a voice confirmation of her vote choices and if she agrees with them, she can finish the voting part of the ceremony by touching the designated part of the screen.

As before, the EBM prints her reciept, including the SerialNumber, division, preferences, and Private WBB signature.

This voter is unable to perform by sight the crucial check that the printed values match her intended vote. Hence she may take both her candidate list and printed preferences to another EBM, which scans the QR code and the printed preferences, and reads her vote back to her. This service is called "The Readback App." It can also read back just a preferences list or just a candidate list.

> **It is essential for integrity that vision-impaired voters are encouraged to check their printed preferences using an independent machine.**

If a compromised EBM can predict that a particular voter will not check their preferences, the vote can be manipulated. This cast-as-intended verification mechanism is requires

---

[2]See http://www.eca.gov.au/research/files/telephone-voting-standard-reviewed.pdf We have also made a new kind of interface, similar to (but invented before) this: http://appleinsider.com/articles/13/03/26/ apple-patents-no-look-multitouch-user-interface-for-portable-devices

trusting that the voter can find at least one EBM in the polling place that does not collude with the first one she used.

An alternative design would be to allow voters to bring their own devices in to perform this check, but this would violate vote privacy because the device might record the data, hence telling someone else how the person voted.

By this point we can be confident that the printed preferences match the voter's intentions. She must now destroy the candidate list.

> **It is essential for privacy that all voters are required to shred their candidate list before leaving the polling place.**

As already mentioned, the EBMs can also speak the preference orders on the slip so the voter can note them down (with a blind note-taker device or with memory). This helps the voter to check the EBM unassisted but does not really affect privacy or verifiability because she must still check that her vote is printed as she requested, and recorded on the WBB as it is printed, rather than trusting the EBM to tell her the truth. She could do the WBB check with assistance from a print reader or from a sighted person without jeopardising privacy.

Note that the only steps that need to be private are the ballot marking by the EBM and check with a second EBM. All the other verification steps: confirmation of the ballot, confirmation of the receipt signature and of correct posting of the receipt to the public WBB, are exactly the same as those for typical voters, and can be performed with assistance without jeopardising ballot privacy.

Confirmation

If she has performed a confirmation check on a ballot, the voter can still go home and use her screen- or print-reader, with the same confirmation-checking software as everyone else, to make sure her candidate list matches the encrypted list on the WBB. The only important detail is that she has to make sure she knows what the cleartext candidate order is. She must either ask several people or use (a) print reader(s). Neither of these impacts upon privacy: there are no privacy implications for anyone in confirming ballots.

3.3. Observing that the vVote output matches what is input into the count

VEC procedures require vVote ballots to be printed out before being incorporated into either the manual tally of paper votes (for LA ballots) or the manual data entry of paper votes into the electronic STV count (for LC ballots). The scrutineers who observe the manual tally must reconcile these ballots with those output from vVote on the public WBB.

The printouts will be visually distinct from ordinary paper ballots. All the vVote votes will bear a unique number on their footer which aligns with a verifiable output vote on the WBB so that they can be checked independently later. Note that these unique numbers

are added to the votes *after* they have been shuffled and hence disassociated from the voter who cast them.

4. System Component details

4.1. The Web Bulletin Board

A number of voting schemes require some form of append-only Web Bulletin Board (WBB). However, specific details of how to design or implement such a service are often lacking. In this section we do not aim to propose a generic WBB, only to define one that will work within the constraints we have and offer the properties we need. The fundamental requirements we have of a WBB are

- that every observer gets the same information, and

- that the data written to it cannot be changed or deleted without detection.

In prior work on the Prêt à Voter protocol, a great deal has been expected of the WBB. It has been expected to prove that it satisfies the properties above, while also responding in real time with signatures and confirmation check information. The design presented here separates those two sorts of roles, breaking the WBB into two:

The Public WBB This is a static digest of the day's transcript. It is updated very infrequently (e.g. once per day). A hash is broadcast via some other channel (e.g. a newspaper or radio broadcast) so everyone can be confident they all get the same information. This data could be replicated extensively because no secret information is held. This is what voters consult for evidence that their votes were included in the tally, and evidence that the tally was correct. The main property is: *A corrupt WBB cannot falsify its data undetectably except by violating some computational assumption.*

The Private WBB A robust secure database which receives messages, performs basic validity checks, and returns a signature. Its correctness based on threshold assumptions. This is implemented as a collection of peers who share a signing key. Validly signed messages are guaranteed, under threshold assumptions, to appear subsequently on the Public WBB. A malicious threshold can collude to misrecord and expose votes. Such misbehaviour is detectable in principle by observing the public WBB, but may not necessarily be provable.

The result of this split should be that the Private WBB can follow a protocol for distributed secure databases without needing to worry about reliable broadcast to the public. The real broadcast channel with memory is a static data structure, the WBB, which is much easier to design. In practice the WBB could be replicated in the cloud. The Private WBB would receive communication only from other "inside" entities such as print servers and EBMs.

4.1.1. Public Web Bulletin Board (Public WBB)

The Public WBB is an authenticated public broadcast channel with memory. We assume some genuine public broadcast channel that can be used to send a small amount of information, specifically a signed cryptographic hash of the transcript. When someone checks the public WBB for inclusion of their data, they also re-hash the contents and check the result against the publicly broadcast one.

The public WBB consists of static data broken into separate commits, with the signed hash of the prior commit step being included in each commit along with the other election data.

Verifying that something appears on the public WBB is a two step process:

- Firstly the observer requests an index file that lists all ballot serial numbers (or whatever other data is being requested) and which commit they are in. The client side code allows the voter to look up the serial number and then download the relevant commit.

- The second step is downloading all the data for the relevant commit, checking that a particular data item (e.g. a particular voting receipt) is present, and checking that the recomputed hash matches the signed, published one.

If the total amount of data per commit becomes large in future, the process of proving inclusion could be made more efficient using hash trees or other log-size data structures [GTT11].

4.1.2. Private Web Bulletin Board (Private WBB)

A private WBB is a robust distributed database which:

- accepts items to be posted (if they do not clash with previous posts),

- issues receipts (which are signed accepted items), and

- periodically publishes what it has received on the public WBB.

The data published on the public WBB for any particular period must include all items that had receipts issued during that period. Robustness is achieved through the use of several peered servers which cooperate on accepting items, issuing receipts, and publishing the bulletin board. They make use of a threshold signature scheme which allows a subset of the peers at or above a particular threshold to jointly generate signatures on data. The peers collectively provide the bulletin board service as long as a threshold of them are honest, and as long as a threshold of them are involved in handling any item posted to the bulletin board. Thus the implementation is correct in the presence of communication failures, unavailability or failure of peers, and also dishonesty of peers. The threshold t required to achieve this must be greater than two-thirds of the total number n of peers: $t > 2n/3$. There is no single point of failure: the system can tolerate failure or non-participation of any component, as long as a threshold of peers remain operational at

any stage. It also allows for different threshold sets of peers to be operational at different times. For example, a peer may be rebooted during the protocol, thereby missing some item posts, and may then resume participation. Details of the protocol are given in a separate publication [CS14].

In addition to robustness and non-repudiation, the private WBB must also perform some basic validity checking on submitted items. In particular, it must reject items that "clash" with previously accepted submissions: a new request to run a confirmation check or vote on a ballot of a given serial number must be rejected if that serial number has already been voted on.

The key properties required of the Private WBB are:

(bb.1) only items that have been posted to the bulletin board may appear on it;

(bb.2) any item that has a signed receipt issued must appear on the published bulletin board (*i.e.* public WBB);

(bb.3) two clashing items must not both appear on the bulletin board;

(bb.4) items cannot be removed from the bulletin board once they are published.

It follows from bb.2 and bb.3 that if two items clash then receipts must not be issued for both of them.

The bulletin board provides a protocol for the posting of an item by an EBM, and its acknowledgement with a receipt. It also provides another two related protocols for the publishing of the bulleting board: an optimistic one, and a fallback. These protocols are given in [CS14] together with their proofs of correctness with respect to the key properties.

4.2. Print-on-demand printers and Randomness Generation Service

The processes for printing and confirming correctness of ballot forms are vital components of Prêt à Voter. This project necessitated a completely new scheme, which is described below.

4.2.1. Protocol overview

Our protocol has two roles. The "randomness generation servers," of which a threshold of at least one are trusted for privacy, send randomness to a "printer". The "printer" uses only that randomness to generate the ballots, which it can then print on demand. In brief:

Before the voting period:

1. Each randomness generation server generates some randomness, commits to it publicly, and sends the opening secretly to the printer.

2. The printer uses the combined randomness to generate the encrypted ballot, which it publishes.

During the voting period:

3. When required, the printer prints the next ballot in sequence, with human-readable candidate names.

There are thus two important points for public confirmation checking:

1. A confirmation check of the encrypted ballot produced in step 2, to check that the candidate ciphertexts are valid and that the printer used the proper randomness. This is described in Section 4.2.2.

2. A standard Prêt à Voter confirmation check of the printed ballot from step 3, to check that the printed human-readable candidate names match those of the encrypted ballot. This is described for our scheme in Section 4.2.3.

Throughout this document when we refer to the "Printer" we are in fact referring to the tablet device that is connected to the printer. As such, the "Printer" has the processing power you would expect to find on a mid-range tablet.

4.2.2. Ballot Generation

The main idea is that the printer generates a permuted list of candidate ciphers using randomness values generated by a distributed set of peers. As such the printer undertakes the expensive crypto operations, but does not have any influence over the values used in those operations. This prevents the printer from mounting kleptographic attacks or otherwise having any influence over the ciphertexts.

During ballot generation the printer is checked to ensure that it has performed honestly: if a sufficient number of ballots are confirmed and shown to be correct then we can gain a high assurance that the printer has behaved honestly. The definition of "honest" is different from standard versions of Prêt à Voter, in which a dishonest printer can only misalign the printed candidate names with the ballot ciphertexts. In our version, a dishonest printer may also attempt to generate invalid ballot ciphertexts or perform a kleptographic attack by using randomness other than that specified by the protocol. However, these two kinds of cheating can be detected by a ballot-generation confirmation check—see below.

Notation:

$Enc_k(m; r)$ is the encryption of message m with public key k and randomness r.[1]

$Dec_k(m)$ is a decryption of m using the private key k.

$ReEnc(\theta; r)$ is a re-encryption (re-randomisation) of the ciphertext θ using the randomness r (this abstracts the requirement of knowing the public key, which is a requirement for ElGamal re-encryption).

---

[1]The Victorian project uses Elliptic Curve El Gamal.

$c(m)$ is a perfectly hiding commitment to message m (using some randomness not explicitly given).

$c(m;r)$ is a perfectly hiding commitment to message m (using randomness r).[2]

PK_E is the election public key (which is thresholded).

PK_P is the printer's public key (which is not thresholded/distributed).

SK_P is the printer's private key.

n is the number of candidates.

b is the number of ballots to be generated for each printer.

G is the number of randomness generation servers.

$RGen_i$ is the i-th randomness generation server.

$SymmEnc_{sk}(m)$ is a symmetric-key encryption of m under the symmetric key sk.

$h(m)$ is a cryptographic hash of the message m.[3]

$SymmDec_{sk}(m)$ is a symmetric-key decryption of m using key sk.

RT_i is $RGen_i$'s private table of encrypted random values (Fig 4.5).

CRT_i is $RGen_i$'s public table of commitments to the values in RT_i (Fig 4.6).

We will post on the public WBB values that are encrypted with a threshold key, or perfectly hiding commitments. We will not post values that are encrypted with a non-thresholded key. We could have used computationally hiding commitments or encryptions with non-thresholded keys, but either of these would have meant that a single leak of relevant parameters, even quite a long time in the future, could have been combined with WBB data to violate ballot privacy. Our system does not achieve everlasting privacy, but it achieves a somewhat related weaker property, that no single entity's data (apart from the printer's) is enough to break ballot privacy, even given WBB data.

Pre-Ballot Generation

Before the ballot generation starts the following must occur:

---

[2]In the vVote project we use the hash-based commitment scheme described in [JJR02].

[3]We use 256-bit AES and SHA-256 respectively. This means that the computational difficulty of guessing a key (2^{256}) is much greater than that of finding a collision (2^{128}). However, this seems justifiable since collision-finding is only useful for cheating during ballot generation, which must be performed in a restricted time, while guessing the symmetric key can be used to break ballot privacy long after the election.

Candidate Name	ID
Vladimir Putin	$cand_1$
Mohamed Morsi	$cand_2$
...	...
Mahmoud Ahmadinejad	$cand_n$

Figure 4.1.: Initial Ballot Input: Candidate Identifiers

PrinterA:1	...	PrinterB:1	...	PrinterC:1
PrinterA:2	...	PrinterB:2	...	PrinterC:2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
PrinterA:b				

Figure 4.2.: Initial Ballot input: Serial numbers for printers A,B,C.

1. The election public key sharers jointly run a distributed key generation protocol to generate a thresholded private key and joint public key PK_E.[4]

2. A list of candidate identifiers is generated and posted on the public WBB, as shown in Figure 4.1. Candidate identifiers are arbitrary, distinct elements of the message space of the encryption function.[5]

3. For each printer, a list of serial numbers of the form "PrinterID:index" is deterministically generated and posted on the public WBB. This serial number is just the literal string as given. These serve as row indices for later computation.

4. Each printer constructs a key pair and publishes the public key PK_P.

All this data, which is posted on the WBB immediately before ballot generation, is shown in Figures 4.1 and 4.2. The protocol for postsing a file to the WBB is shown in Figure 4.3.

The following sections describe the process of ballot generation for a single printer, but it should be clear how the same process will be run in parallel for each printer.

Randomness Generation

The randomness generation consists of each server $RGen_i$ generating a large table of secret random values and sending them (privately) to the printer after posting (public) commitments to them on the WBB. The protocol for posting the public commitments is illustrated in Figure 4.4.

---

[4]We keep the key sharers and the randomness generation servers conceptually separate, even if we end up using the same servers.

[5]For vVote, candidate identifiers are elliptic curve points either randomly selected or calculated as part of the optimisation used to speed up mixing, depending on the type of race.
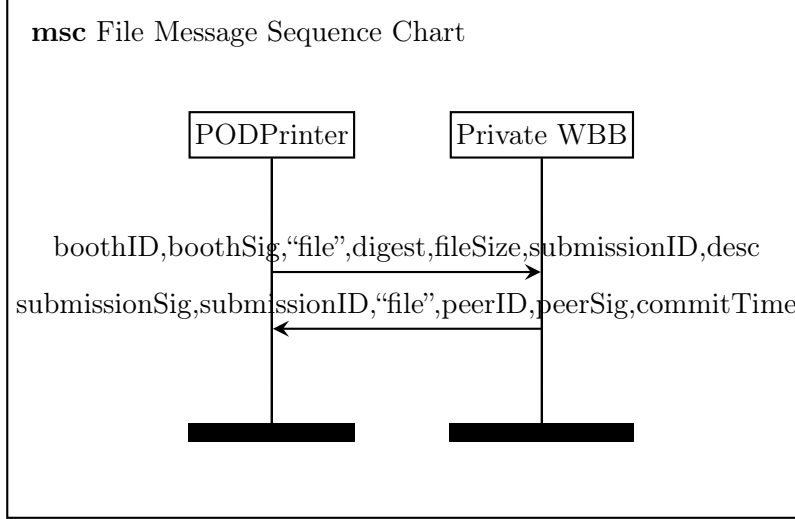
- boothSig : $\mathrm{Sign}_{PODPrinter}\{\mathrm{serialNo,digest}\}$

- peerSig : $\mathrm{Sign}_{WBB}\{\mathrm{submissionID,digest,senderID,commitTime}\}$

Figure 4.3.: File Message Sequence Chart

**Detailed algorithm** In detail: each random value has length k, where k is a security parameter which should be about 256 bits. Each server $RGen_i$ generates a random symmetric key sk_i. It then generates a table of $b * n$ pairs of pieces of random data, each of size $2 * k$ bits and encrypted under sk_i. The table is denoted by RT_i, and each pair can be retrieved by the serial number and column, or the row and column.[6] The result is shown in Figure 4.5. The idea is that the first element of each pair, $r_{(row,col)}$, will be used later by the printer; the second element, $R_{(row,col)}$, is used to commit to $r_{(row,col)}$ and to open the commitment in case of confirmations.

Each server commits to $r_{(row,col)}$ by posting on the WBB a commitment to it using randomness $R_{(row,col)}$. The table of commitments is shown in Figure 4.6. Call the table CRT_i. Each peer $RGen_i$ posts its CRT_i, and checks all CRT_i's are posted before sending RT_i privately to the printer.[7] $RGen_i$ also encrypts sk_i with the printer's public key and sends the result (denoted $esk_i = Enc_{PK_P}(sk_i; r)$) to the printer.

---

[6] For example $RGen_i.RT_i(PrinterA:1,2)$ and $RGen_i.RT_i(1,2)$ will return the pair of encrypted random values in the second column for the first ballot: $SymmEnc_{sk_i}(r_{1,2}||R_{1,2})$.

[7] This is to stop the last peer choosing their randomness when they know the others'. If this was not enforced, then one bad randomness generation server colluding with a printer could determine the randomness values for each of that printer's ballots, thus breaking privacy. The bad server would wait until the printer told it all the other servers' random values, then generate its own to produce a particular final value.
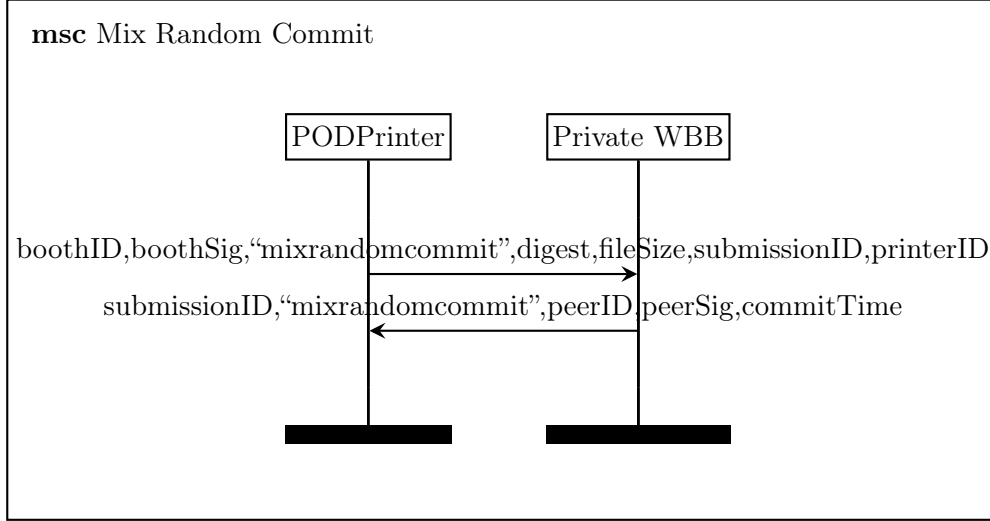
- boothSig : $\mathrm{Sign}_{PODPrinter}\{$submissionID,printerID,digest$\}$

- peerSig : $\mathrm{Sign}_{WBB}\{$submissionID,senderID,printerID,digest,commitTime$\}$

Figure 4.4.: Mix Random Commit Message Sequence Chart

Ballot Permutation and Commitment

The printer receives the respective RT_i and corresponding (encrypted) key esk_i from each server. The printer also downloads or constructs the candidate identifiers and serial numbers shown in Figures 4.1 and 4.2. The printer now needs to encrypt and permute the candidate identifiers.

For each candidate identifier $cand_k$ in the pre-committed table in Figure 4.1 it encrypts it using the combined randomness from the RT tables, received from the randomness generation servers. To combine the randomness the printer first decrypts the encrypted symmetric key esk_i received from each server and then uses the resulting key sk_i to decrypt the randomness in RT_i. For each element of each RT, the printer checks that the decrypted pair $r_{row,col}, R_{row,col}$ opens the commitment at $CRT_{row,col}$. It challenges any that do not—see below for what it should do when it detects $RGen_i$ cheating at this point.

The decrypted first elements $r_{row,col}$ from each peer are concatenated and hashed. The printer then uses the hash output for randomness when encrypting the candidate identifier under PK_E. The resulting ciphers are then sorted into canonical order to produce a random permutation. These ciphers are posted on the public WBB. Note that the output of the encryption is pseudo-random and as such sorting the encrypted ciphers will give a pseudo-random permutation π. The printer retains this permutation so that it can print the plaintexts in the appropriate order when requested to print that ballot. After the ciphers are submitted to the WBB, the confirmation checking protocol detailed

SerialNumber	Encrypted Randomness			
PrinterA:1	$SymmEnc_{sk_i}(r_{1,1}\|R_{1,1})$	$\ldots$		$SymmEnc_{sk_i}(r_{1,n}\|R_{1,n})$
PrinterA:2	$SymmEnc_{sk_i}(r_{2,1}\|R_{2,1})$	$\ldots$		$SymmEnc_{sk_i}(r_{2,n}\|R_{2,n})$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
PrinterA:b	$SymmEnc_{sk_i}(r_{b,1}\|R_{b,1})$	$\ldots$		$SymmEnc_{sk_i}(r_{b,n}\|R_{b,n})$

Figure 4.5.: Ballot Input: Table RT_i, sent privately from peer i to printer A without public posting.

SerialNumber	Committed Randomness			
PrinterA:1	$c(r_{1,1}; R_{1,1})$	$\ldots$		$c(r_{1,n}; R_{1,n})$
PrinterA:2	$c(r_{2,1}; R_{2,1})$	$\ldots$		$c(r_{2,n}; R_{2,n})$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
PrinterA:b	$c(r_{b,1}; R_{b,1})$	$\ldots$		$c(r_{b,n}; R_{b,n})$

Figure 4.6.: Commitment to Initial Ballot Input: Table CRT_i, posted by peer i on the WBB.

in Section 4.2.2 can be run. The Ballot Generation Commitment protocol between the printer and the bulletin board is given in Figure 4.7.


Algorithm 1: Deterministic Encryption by Printer

for $i = 1 \to G$ **do**   ▷ Decrypt the symmetric keys from the $RGen$Servers
    $sk_i \leftarrow Dec_{SK_P}(esk_i)$
end for
for $j = 1 \to b$ **do**   ▷ b is the number of ballots.
    for $k = 1 \to n$ **do**   ▷ n is the number of candidates.
        $rand \leftarrow SHA(SymmDec_{sk_1}(RT_1(j,k))\| \ldots \|SymmDec_{sk_G}(RT_G(j,k)))$
        $CT_{j,k} \leftarrow Enc_{PK_E}(cand_k; rand)$
    end for
    $CT_{(j)} \leftarrow Sort(CT_{(j)})$ ▷ $CT_{(j)}$ returns the entire row
    $perms_j \leftarrow \pi$ ▷ π is the permutation applied to sort CT_j.
end for
Send CT to WBB.

The algorithm run by the printer is given in Algorithm 1. The intention is that only the printer knows which ciphertexts correspond to which candidates, but its algorithm for generating those ciphertexts is deterministic. Hence it cannot use the ciphertexts to leak information without detection. Of course, the printer could always leak that information via a side channel, but this is unavoidable and occurs with any form of electronic ballot printing or marking.

What the printer should do if $RGen_i$ cheats It is important in the above protocol that the printer checks the opening of each commitment, *i.e.* checks that for each element of
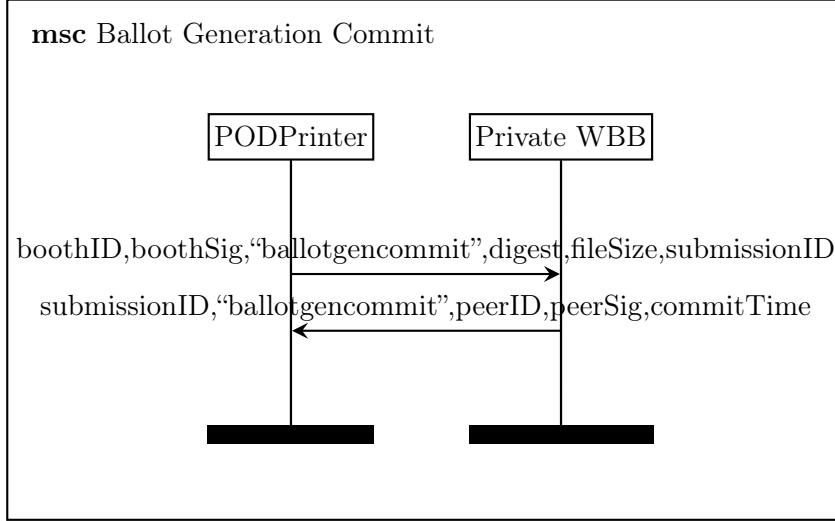
- boothSig : $\text{Sign}_{PODPrinter}\{\text{submissionID,digest}\}$

- peerSig : $\text{Sign}_{WBB}\{\text{submissionID,digest,senderID,commitTime}\}$

Figure 4.7.: Ballot Generation Commit Message Sequence Chart

each RT, the decrypted pair $r_{row,col}, R_{row,col}$ opens the commitment at $CRT_{row,col}$. It is important in practice that the printer raise an alarm on any commitments that are not correctly opened. Exactly how such a dispute should be resolved requires some careful engineering of procedures. It is difficult to tell whether the printer or the randomness generation server is misbehaving without exposing private ballot data. This is not necessarily a problem, because the randomness contributed by other randomness generation servers would not be exposed. Hence the other ballots remain private.

Note that the issue does not affect public verifiability, because the absence of proper commitment opening would be detected by a confirmation check of this ballot. It does, however, affect accountability: if a confirmation check detects that the value used to encrypt a ballot was not a valid opening of the commitment on the WBB, we would like to know whether it was the randomness generation server or the printer that cheated. If we insist that the printer performs this check, then we can be certain that a failed confirmation is the printer's fault.

Alternative construction with PRNGs   Rather than generate a separate random value for each candidate in each ballot, an alternative is to generate one random value for each ballot, then use a cryptographic Pseudo-Random Number Generator (PRNG) to expand it to produce randomness for all of the encrypted values in the ballot. This introduces an assumption on the good expanding behaviour of the PRNG, but substantially reduces the communication costs of the protocol. Hence the tables of Figures 4.5 and 4.6 would

need only 2 columns rather than $n + 1$. It does not significantly change confirmation checking.

However, we have not taken this approach for the Victorian system due to practical concerns about PRNG implementations. The most important is the possibility that variations in implementations could imply a failure of reproducibility of the random sequences, which is required for confirmation checking. In principle this should not be a difficult issue to address. In practice we were concerned this would make it significantly harder to write an independent verifier, so we opted to omit the PRNGs.

Ballot Generation Audit

A suitable percentage of ballots are chosen at random for audit.[8] For each ballot selected, the printer posts on the WBB the randomness it used during the generation, *i.e.* to open the commitments for that SerialNumber in each peer's CRT table. The protocol is shown in Figure 4.8. The printer can either have this stored, or else can recalculate it from the encryptions it received prior to ballot generation. Anyone can verify the commitment openings $(r_{row,col}, R_{row,col})$ and reconstruct the ballot ciphertexts from them. Thus anyone can check that the ballots were correctly constructed and that the printer used the appropriate randomness.
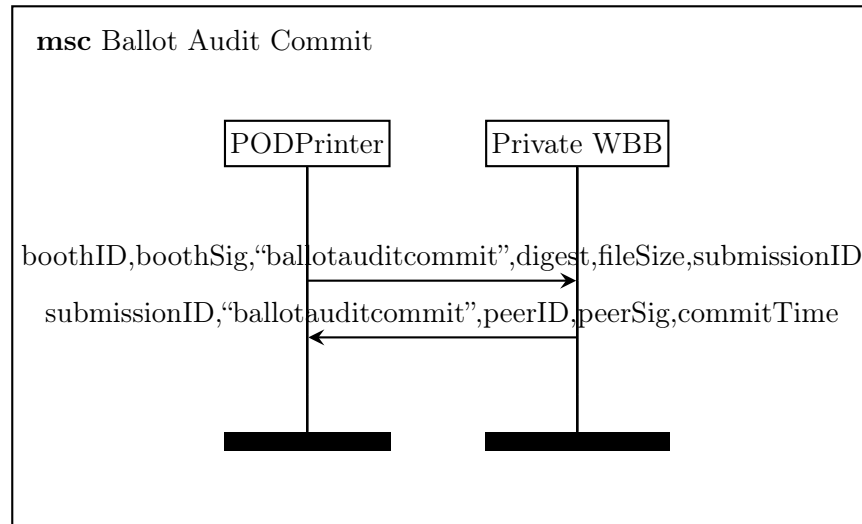
- boothSig : $\text{Sign}_{PODPrinter}\{\text{submissionID,digest}\}$

- peerSig : $\text{Sign}_{WBB}\{\text{submissionID,digest,senderID,commitTime}\}$

Figure 4.8.: Ballot Audit Commit Message Sequence Chart

---

[8]The questions of who chooses, how they choose, and how it can be guaranteed that they choose well enough to engender confidence in a particular election result are discussed in Section 6.1.2.

4.2.3. Print on Demand

This section describes what happens when a voter appears at a polling place. The printer needs to print a pre-generated ballot for the appropriate district. The printer knows the plaintexts and permutation for a particular ballot so can easily print the appropriate ballot out.

However, there is a risk that a misbehaving printer might print a completely invalid ballot, *i.e.* one that has not been part of the generation process described above. Although this is detectable by confirmation checks, we prefer for practical reasons to prevent it altogether. Hence we require that the printer obtain a signature from the WBB in order to create an authentic ballot. The WBB is attesting to those ciphertexts matching what the printer has already committed to.

The vVote project uses a combined EBM and scanner rather than the traditional Prêt à Voter technique of filling in the ballot with a pencil and then scanning it. The print on demand protocol works exactly the same either way, but the description below assumes the vVote-style combined scanner and EBM.
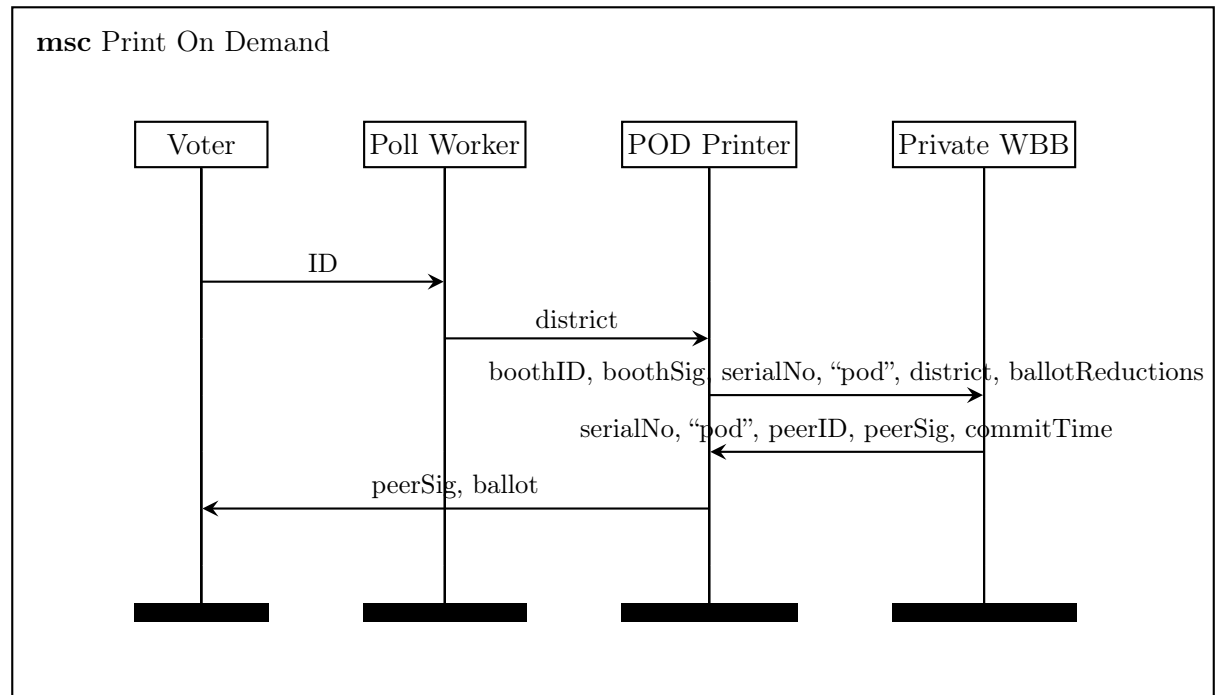
- boothSig : Sign$_{PODTablet}${serialNo,district}

- peerSig : Sign$_{WBB}${serialNo,district} (the returned peerSig forms the serialSig in future sequence charts)

Figure 4.9.: Print on Demand Message Sequence Chart

Figure 4.9 shows the message sequence chart for print on demand, which is elucidated in Section 4.2.3 below. The authorisation, ballot reduction and serial number signing all take place together in a single round of communication. The signatures generated are deterministic BLS signatures [BLS01], including the signature from the WBB.

Ballot Reduction

It is unpredictable exactly how many of each ballot will be required at each location. We could have generated an abundant oversupply of ballots with exactly the right number of candidates for each division, but this would have been quite expensive. Instead, for efficiency, we generate an abundant oversupply of generic ballots with a larger than necessary number of candidates, then reduce it down to the appropriately sized ballot for the district/region it is going to be used in. This allows great flexibility about who votes at what polling place—any voter can arrive anywhere and have a ballot produced to match their voting eligibility.

If the ballot contains more ciphers than candidates, we need to reduce the ballot in a manner that can be verified. The following proposal has the nice feature that the voter (or the EBM if there is one) does not need to know where the blanks are in order to cast the vote: they just get a permuted list of the candidates they were expecting. Then the EBM prints, and the voter checks, the voter's preference numbers against that order.

Suppose from now on there are m candidates in the division and n $(> m)$ ciphertexts on the ballot. ($n = m$ is a special case of the steps below.) The printer is supposed to use the ciphertexts for $cand_1, cand_2, \ldots, cand_m$. Of course it could cheat and attempt to use other ciphertexts instead, but this could be detected at confirmation checking like any other kind of bad printing. We want to be able to demonstrate afterwards on the WBB that it used the right ciphertexts. The protocol is as follows:

Pollworker: authenticates the voters (using whatever secure or insecure method is traditional) and sends a print request to the Print On Demand device specifying the district/region they can vote in,[9]

Printer: retrieves the next available ballot and looks up the number m of candidates in the submitted district/region.

Printer: sends to the WBB:

- the SerialNumber,
- the division, and
- a list $BallotReductionRandomness$ of randomness values for the unused ciphertexts (i.e. the ones from $cand_{m+1}$ to $cand_n$), together with their respective permuted locations so the WBB can check them. The randomness values are those computed by the printer in the algorithm in section 4.2.2)

---

[9]More generally, it is the pollworkers' responsibility to authenticate the voter and request the ballot(s) that the person is eligible to vote on.

WBB: checks that the ciphers held for $cand_{m+1}$ to $cand_n$ are encryptions of the candidate IDs of the unused candidates for the specified division.

- if valid it signs the serial number and the division and returns it to the printer, and posts the randomness values to the WBB so they can be publicly checked;

- if invalid it returns an error message.

Printer: Checks the WBB signature of the serial number and division and, if valid, prints the ballot and signature. The printer knows the permutation and plaintexts so does not need to do any crypto to print the ballot

Voter: votes on the ballot exactly as if it had been generated for the right number of candidates,

Voter: shreds the candidate list,

EBM: submits the ballot to the WBB exactly as if it had been generated for the right number of candidates,

WBB: accepts (and signs) the ballot only if it is accompanied by a signed serial number and division

EBM: prints the sig on the receipt,

Voter: (optionally) checks the sig, which covers only data visible to the voter.

Voter: later checks their vote on the WBB. They only need to check the serial number and order of their preference numbers—the correct opening of the unused (too big) candidate numbers will be universally verifiable.

Print confirmation

Suppose a voter wants to confirm a printed ballot, i.e. to check that the printed candidate list matches the ciphertexts on the WBB. The following is performed:

Voter: requests a confirmation check from the same printer that printed their ballot,

Printer: sends to the WBB the randomness to open the commitments to the randomness on each CRT_i used to generate the ballot in the ballot generation phase

WBB: checks the serial number has not already been voted on or confirmed and if not, opens the commitments, reconstructs the ballot, computes the permutation π, posts all the data on the public WBB, and sends a jointly signed copy of π (or candidate names in permuted order) to the printer

Printer: The printer prints the signature

Voter: checks the signed order of candidates π against the order printed on the ballot (note, the permutation signed by the WBB should reflect any successful ballot reduction already performed).

Voter: takes their confirmed ballot home and checks that the value provided on the WBB matches the candidate order that was signed.

msc Confirmation Check

Voter | POD Printer | Private WBB

serialNo, district, serialSig, permutation

boothID, boothSig, serialNo, serialSig, permutation, commitWitness, "audit", district

serialNo, "audit", peerID, peerSig, commitTime

serialNo, district, serialSig, permutation, peerID, peerSig, commitTime
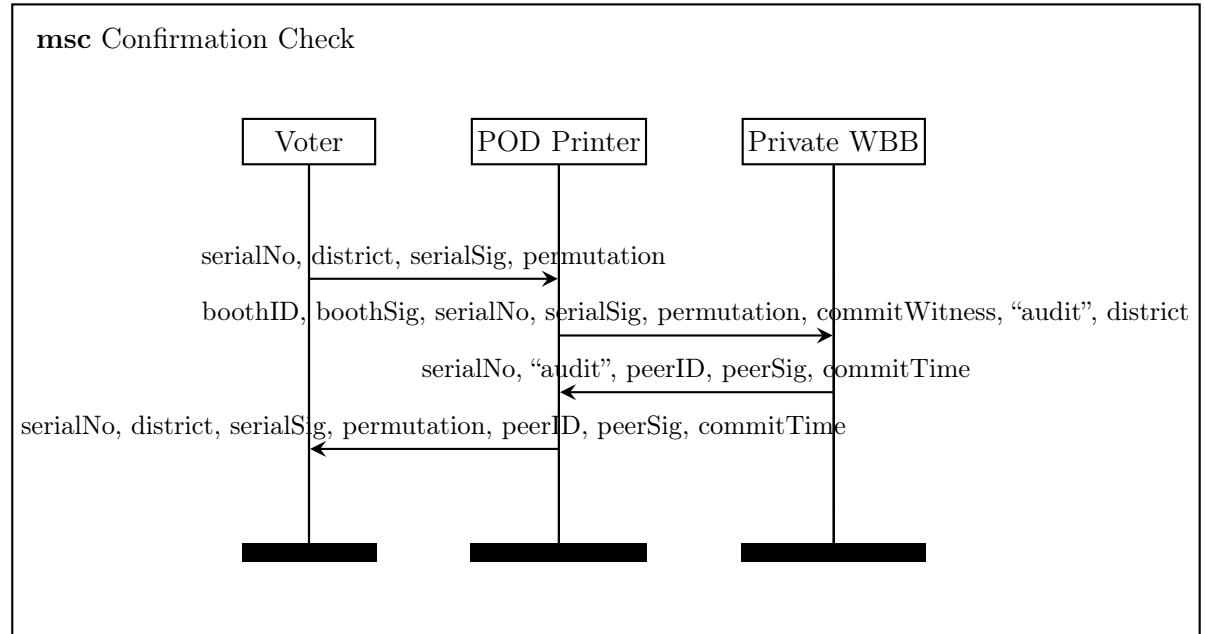
- boothSig : Sign$_{PODTablet}${serialNo,"audit",permutation,commitWitness}

- serialSig : Sign$_{WBB}${serialNo,district}

- peerSig : Sign$_{WBB}${"audit",serialNo,reducedPermutation,commitTime}

Figure 4.10.: Print on Demand Confirmation Check Message Sequence Chart

Figure 4.10 shows the message sequence chart for confirming.

Forward Secrecy

The randomness held on a printer is sufficient to reconstruct the ballots, and hence reveal the candidate orderings. If a printer is stolen, the randomness it holds will expose the associated ballot forms. Hence it is desirable for a printer to delete the randomness for ballots on which votes have been cast, since there is a potential privacy breach.

However, when a printer prints a ballot it must allow for a print confirmation check which will require it to open the commitment to the randomness. Therefore, after a printer has printed a ballot, it must retain this randomness for some period of time.

After the ballot has been used to cast a vote, a confirmation check is not allowed and so the randomness no longer needs to be retained and can be deleted. Deletion can be triggered by a confirmation message to the printer, signed by the WBB, when a

vote is cast. Alternatively a time limit can be set on a confirmation request following ballot printing, and the randomness can be deleted after that time if no request has been received.

Note that the encrypted randomness values and symmetric key are sent directly to the printer and not posted on the WBB. As such, there is no publicly available information that could be combined with a stolen, but previously honest, printer to reveal used ballots. The symmetric keys sk_i should be deleted from the printers after the RT_i tables have been decrypted.

A faster variant with a shorter permutation commitment

Although the above protocol is quite efficient, it still requires the WBB to do a lot of computation to open all relevant commitments and reconstruct the ballot permutation each time a print confirmation check occurs. This is unfortunate because we would like to encourage ordinary voters to perform print confirmations by making them easy and fast.

One way to speed up print confirming is to ask the printer to commit to the candidate-list permutation π directly when it generates the ballot, then ensure that this commitment to π is confirmed for proper generation (during ballot generation confirmations) and for conformance with the ballot permutation (during print confirmations). During a print confirmation check, the WBB needs only to open and verify the commitment to π, then sign it and return it to the printer. Every print confirmation check then triggers a ballot generation confirmation check, which opens all the commitments just as described in Section 4.2.2, but this does not have to be done while the voter is waiting for the print confirmation check to complete.

Like other randomness values used by the printer, the randomness used in the commitment must also be generated by the $RGen$ servers, using a new column in each RT_i and CRT_i table. The printer retrieves the random value the same way that it retrieves all the others, and uses it to compute the commitment to π. That commitment is sent, along with ballot generation ciphers, to the WBB during the ballot generation stage. It is also checked, along with the ciphers, during the ballot generation confirmations so we gain a statistical assurance the commitments to the permutation are correct. During the print confirmation check everything proceeds as described above, except the WBB only has to open and check the commitment and then sign the permutation based on that.

This does not affect universal verifiability, because the same data linking the ballot permutation to the commitments on the WBB is eventually published either way. However, the big advantage is it reduces the workload on the WBB during the critical time that the voter is waiting for the signature on π, since it can now defer the re-encryptions necessary to verify the permutation.

More precisely, if we add the required random values into the $n + 1$-th column of each RT_i, Algorithm 1 would now be:

Ballot generation confrimation check (Section 4.2.2) would be exactly as above. Additionally, anyone can recompute the randomness value the printer used to commit to the candidate permutation π, and hence open that commitment and check that it matches

43

Algorithm 2: Deterministic Encryption by Printer with explicit WBB commitment to π.

---

for $i = 1 \rightarrow G$ **do** $\triangleright$ Decrypt the symmetric keys from the *RGen*Servers
    $sk_i \leftarrow Dec_{SK_P}(esk_i)$
end for
for $j = 1 \rightarrow b$ **do** $\triangleright$ b is the number of ballots.
    for $k = 1 \rightarrow n$ **do** $\triangleright$ n is the number of candidates.
        $rand \leftarrow SHA(SymmDec_{sk_1}(RT_1(j,k))\| \ldots \|SymmDec_{sk_G}(RT_G(j,k)))$
        $CT_{j,k} \leftarrow Enc_{PK_E}(cand_k; rand)$
    end for
    $CT_{(j)} \leftarrow Sort(CT_{(j)})$ $\triangleright$ $CT_{(j)}$ returns the entire row
    $perms_j \leftarrow \pi$ $\triangleright$ π is the permutation applied to sort CT_j.
    $rand2 \leftarrow SHA(SymmDec_{sk_1}(RT_1(y,n+1))\| \ldots \|SymmDec_{sk_G}(RT_G(y,n+1)))$
    $commit_\pi \leftarrow c(\pi; rand2)$
end for
Send CT and $commit_\pi$ to the WBB. to WBB.

---

the ballot permutation.

Print confirmation (Section 4.2.3) would be:

Voter: After collecting printed candidate list, remains at printer and requests a confirmation check,

Printer: sends to the WBB the randomness to open the commitments to the randomness on each CRT_i used to generate the ballot in the ballot generation phase and the randomness to open the commitment to π.

WBB: (immediately) checks the serial number has not already been voted on or confirmed and if this is true, opens the commitment to π, checks it, and if valid sends a jointly signed copy of π (or candidate names in permuted order) to the printer.

WBB: (later) opens all the commitments for this ballot, reconstructs the ballot, computes the permutation π, posts all the data on the public WBB.

The final step of opening all the other commitments reduces the total number of confirmations that need to be done. If we relied entirely on the immediate check of the serial number and the frequency of ballot generation confirmations, then the system would still be universally verifiable, but the probability of the printer cheating successfully would be higher.

Real-time Printer Replacement

If a printer is stolen or fails, all the ballots that had been generated by it are no longer available for use. This is important because the "printer" is a small tablet PC that would be easy to carry. Hence we need to have a suitable method for bringing replacement equipment back online during election time. For example, we could bring the randomness

generation servers back online each night, or when needed, to generate new randomness, after having deleted those values they had already sent to printers. An alternative is for the randomness generation authorities to do the same thing as described above for a few extra as-yet-undeployed printers, put the data on the public WBB, and then ask each one to send their sk_i to some (distinct) entity who is going to be online at voting time.

4.3. Electronic Ballot Marker (EBM)

The EBM is a computer that assists the user in filling in a Prêt à Voter ballot. The device is already adequately described in the Introduction. The protocol for a user to start the EBM to make it ready to receive a vote is given in Figure 4.11; and the protocol for casting a vote once it has been provided to the EBM is given in Figure 4.12.
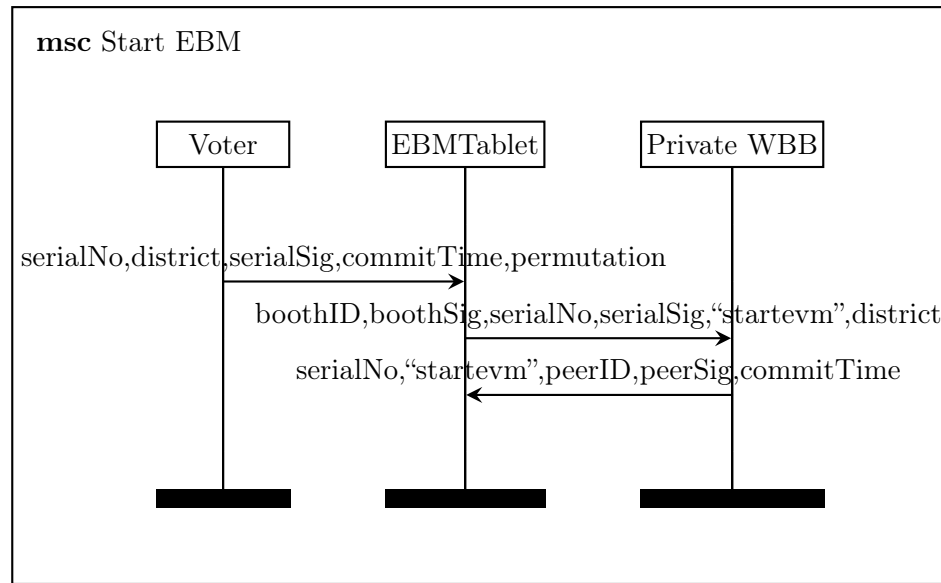
- boothSig : $\mathrm{Sign}_{EBMTablet}\{$"startevm",serialNo,district$\}$

- serialSig : $\mathrm{Sign}_{WBB}\{$serialNo,district$\}$

- peerSig : $\mathrm{Sign}_{WBB}\{$"startevm",serialNo,district$\}$

Figure 4.11.: Start EBM Message Sequence Chart

msc Vote Message Sequence Chart

| Voter | EBMTablet | Private WBB |

serialNo,district,serialSig,commitTime,permutation

races,boothID,boothSig,serialNo,serialSig,"vote",startEVMSig,district

serialNo,"vote",peerID,peerSig,commitTime

serialNo,district,serialSig,permutation,peerID,peerSig,commitTime
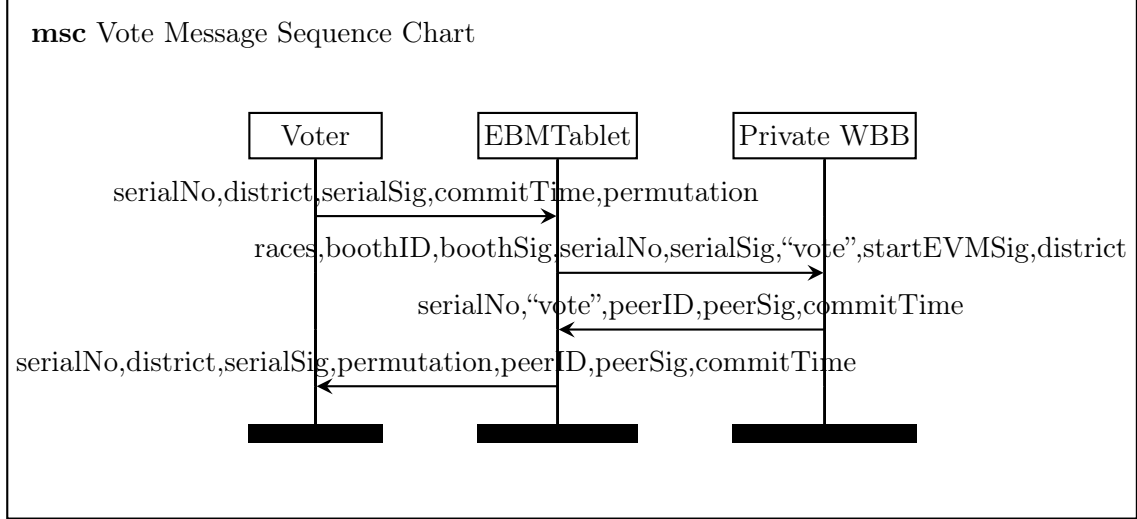
- boothSig : $\text{Sign}_{EBMTablet}\{\text{serialNo,district,preferences}\}$

- serialSig : $\text{Sign}_{WBB}\{\text{serialNo,district}\}$

- startEVMSig : $\text{Sign}_{WBB}\{\text{"startevm",serialNo,district}\}$

- peerSig : $\text{Sign}_{WBB}\{\text{serialNo,district,preferences,commitTime}\}$

Figure 4.12.: Vote Message Sequence Chart

4.4. Cancel Station

The Cancel Station is a supervised interface for cancelling a vote (identified by Serial Number). This will be implemented on the same devices as the print-on-demand printers, but remains conceptually distinct (and could easily be implemented on a separate device).

To request cancellation of a vote, a voter presents the printed candidate list to the election staff. The candidate list contains the signed serial number, which is scanned by the cancel station and passed to the Cancel Authority to run centrally. That cancel authority constructs its own signature on the cancellation request and sends it back to the Cancel Station. That cancel authorisation is then forwarded to the Bulletin Board peers for inclusion on the bulletin board. A cancellation receipt is returned for the voter to retain, signed by a threshold of the bulletin board peers. This protocol is illustrated in Figure 4.13. A precise description of the surrounding procedures is given below.

Each Cancel Station will have a (low) limit on the number of cancellations that may be submitted through it before being checked by VEC HQ. Hence a malicious Cancel Station incorrectly requesting cancellation of votes will not be able to cancel many before it is under scrutiny. Furthermore the cancel requests may be revoked if they are discovered to be flawed, and the votes can then be reinstated.

4.4.1. Procedures for vote cancellation

It is important to emphasise that a vote is never cancelled unless the elector follows this procedure. A voter who leaves the polling place with a signed receipt must never have it subsequently cancelled. Voters must also not be allowed to retain vote receipts associated with cancelled ballots. This procedure is designed to rectify some problems that might arise during the voting process, cancel the vote, and ensure that the voter's signed receipt for the cancelled vote (if they have one) is marked as cancelled. The process is:

1. A voter complains that their printed preferences do not match their expressed intention.

2. The voter provides the candidate list. If the candidate list is already shredded or missing, then cancellation is refused and the elector can only leave the polling place.

3. Polling official scans the Serial Number on the ballot and requests a quarantine.

4. VEC HQ provides permission for the cancellation to occur. Permission uploaded to WBB, which replies to printer with a receipt. Printer prints a signed cancellation onto the ballot.

5. Voter checks signature on cancellation.

6. Elector can be given a full paper vote (with human assistance) or be given another e-vote.
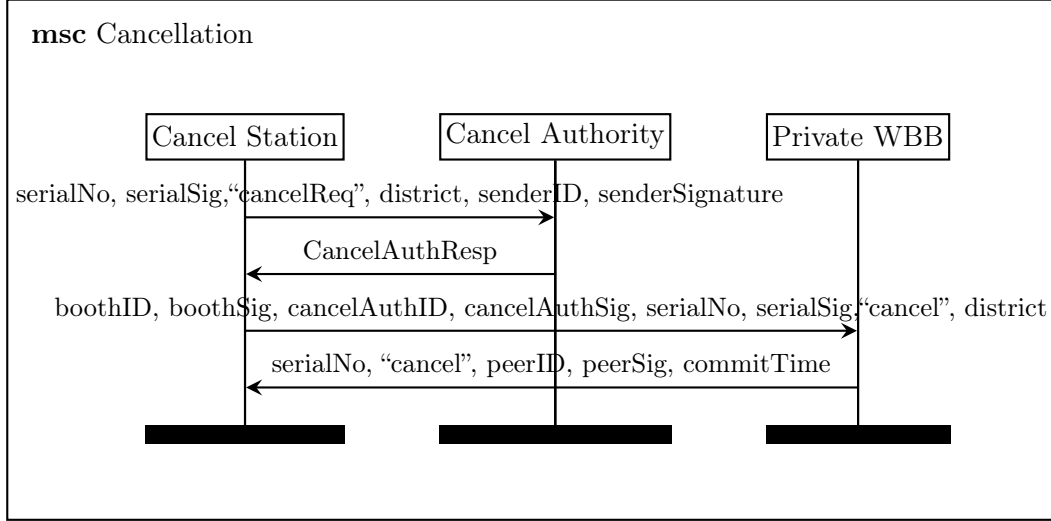
- senderSignature : $\text{Sign}_{VPS}\{\text{"cancelReq"},\text{serialNo},\text{district}\}$

- CancelAuthResp : $\text{Sign}_{CancelAuth}\{\text{cancelAuthID},\text{cancelAuthSig}\}$

- boothSig : $\text{Sign}_{VPS}\{\text{serialNo},\text{"cancel"}\}$

- cancelAuthSig : $\text{Sign}_{VPS}\{\text{serialNo},\text{"cancel"}\}$

- serialSig : $\text{Sign}_{WBB}\{\text{serialNo},\text{district}\}$

- peerSig : $\text{Sign}_{WBB}\{\text{"cancel"},\text{serialNo}\}$

Figure 4.13.: Cancellation Message Sequence Chart

4.5. Mixnet

The system uses a re-encryption mixnet which produces a noninteractive, universally verifiable proof of a shuffle and decryption (of encrypted votes) and posts it to the Public WBB. The main advantage of the re-encryption (as opposed to decryption) mixnet is that it separates the processes of shuffling and decryption. If one mix server blocks, it can simply be removed and the mix re-run. Decryption works as long as a threshold of key share holders remains available.

The current implementation uses Randomised Partial Checking [JJR02] with the modifications proposed by Khazaei and Wikström [KW13], including requiring all mixnet peers to contribute to the randomly generated challenges.

Each division will be shuffled separately.

There will be a problem if the number of votes cast in a particular division (for ATL or BTL) is too small to be an acceptable anonymity set. This is an unavoidable problem that has nothing to do with the choice of mix protocol, though it is affected by the

decision to allow receipts to reveal the choice of ATL or BTL. This means that that data, along with the division name, will be revealed on the mix inputs, so each category needs to be mixed separately. Some categories could be quite small. It will be up to the administrators to decide how small an anonymity set must be before the disadvantages of privacy compromise outweigh the advantages of public verifiability.

We would pad all votes going through the mix to the same length. If someone provides 27 preferences, but everyone else provides 36, we will pad the set of ciphers for the 27 to be the same length. The padding is a zero/null value that is agreed and published in advance. That prevents trivially tracking the data through the mix. However, it does not prevent matching a receipt of 27 preferences to a decryption of 27 preferences. See Section 6.2.5 for a discussion of the options and implications.

5. Robustness and recovery from failures

Section 3 described a series of checks that voters and others can perform to ensure integrity, but did not specify exactly what happens when any of these checks fail. It is challenging in any voting system to recover from errors. A failure of check 1, 3 or 4 is immediately demonstrable (assuming Check 3 is performed on the spot in the polling place) and proves malfeasance by election authorities. This would have serious implications for the trustworthiness of the election result. It is less clear how seriously to regard a failure of Check 2. Unfortunately there will be some rate of false alarms, in which voters claim their vote was misrecorded when they simply misremembered it or changed their minds. Hence a zero-tolerance policy is unworkable, even though any tolerance increases the chances for vote manipulation. Whatever the level of tolerance, it is important that ballots spoiled in this way remain secret, or the process can introduce opportunities for coercion.

One simple fallback method, in the event of unacceptable failures including the loss of network connectivity or complete device failure (*e.g.* a printer or EBM malfunctioning irretrievably), is simply to stop using Prêt à Voter and use the EBM as a plain electronic ballot marker.

5.1. Fallback to Plain EBM mode

The protocol requires the participation of the Private WBB. Intermittent loss of connection to the WBB will be handled by the communication infrastructure, but loss of the network or the WBB will require fallback to a mode of operation that is purely local.

Since vVote is intended to operate alongside standard paper ballots (for 2014) the fallback position will be to return to "plain Electronic ballot marker mode": the EBM machines will be used purely for constructing and printing a paper ballot which can then be cast with the standard paper ballots. This would contain the voter's preferences in boxes beside candidate names (and parties/groups) listed in the offical order, without the serial number or any other identifying information. The result would look as much like the manually-completed paper ballots as possible given the printer available.[1]

No receipts will be issued and the vote will not be submitted through vVote. The Print on Demand service is not used at all for plain EBM voting.

This provides simple cast-as-intended verification, but no more evidence than the ordinary paper system that the vote was properly (transported and) included in the count.

---

[1] Legislation specifies certain formatting rules for ballots, so technically speaking these would not be ballots, but some sort of summary representation of the voter's intention that could be counted manually in place of a ballot.

In other words, this system is software independent but still depends on procedures in the polling place and possibly in the transport of paper votes. This is a reasonable fall-back in the event that vVote is unavailable, particularly for voters who would require assistance to complete a paper ballot manually.

Of course if there is a complete power failure then all voters will have to revert to paper voting.

5.2. Recovery from other failures

The rest of this section describes recovery strategies from failures that are not so overwhelming as to motivate reverting to plain EBM mode. To date little work has been done on robustness, which mainly concerns analysing and providing safeguards against possible system faults. We define robustness of an e-voting system as the ability to tolerate failure of any of its components. On the practical level, recovery from failure/misbehaviour should also be timely.

The importance of adequate recovery measures is highlighted when a system is deployed for the real world. Provision for occurrences such as network or power outages, device malfunction, *etc.*, that do not feature significantly in theoretical work becomes a more pressing requirement. Something as apparently innocent yet all too real as a printer breaking down could, for instance, bring voting to a standstill, no matter how secure the protocol is proven otherwise.

As might be expected for complex systems which require a high level of security and reliability, the task of accounting for all possible failures in an e-voting system is extremely involved, and demands rigorous attention to detail. It is crucial that any authority running an election is ready with appropriate procedures to deal with any deviations from a normal protocol that may occur: from the seemingly mundane, such as printers running out of paper to the more serious, such as errors in device digital signatures.

Officials should keep detailed records of device and procedural failures of any kind. Repeated and/or widespread occurrences, e.g. in the device failing to sign receipts may necessitate de-commissioning and replacing a device, and forensic examination to determine the possible cause. The authority may need to specify a number for each type of failure, after which further investigation is carried out.

We will walk through the vVote voting ceremony and consider the various error states that may arise, and the possible remedial procedures to ensure robustness of the system. Many of the circumstances require a vote to be *cancelled*, which has its own special set of procedures, described in Section 3.1.4.

We assume there is an established procedure for registering and authenticating voters at the polling station, for example marking off the electoral roll on production of some recognised form of identification or claim of identity. This may not be trivial if the register is electronic, in which case it may be affected by a network or power outage (but this is out of scope).

5.3. Potential failures in ballot generation or ballot printing confirmation

The authority should encourage voters and third parties who may wish to confirm ballots during either the ballot construction phase or the voting phase.

Print station: does not work, either for printing or for confirming, or does not print a valid WBB signature on the ballot:

Persistent device failure has been discussed in Section 5.1.

A printer that occasionally fails to produce a proof of correct opening in response to a confirmation request should be treated with suspicion. In the absence of a good reason for expecting network failures (preventing the print station from contacting the private WBB), the assumption should be that the ballot is not properly formed. The threshold for replacing such a printer should be low. A printer that had printed a malformed ballot would not necessarily fail by proving that the ballot was malformed—more likely, it would fail to provide a proof at all.

Ballot is incorrectly formed: Finding inconsistencies in ballot formation could potentially occur during pre-election checking, or during voter-initiated checking during the voting phase.

A printed ballot with a proper WBB signature but a candidate list that is inconsistent with the cryptographic information opened in a confirmation check, represents a serious failure of the system. The affected device should not be used.

5.4. Potential failures in the voting ceremony

After obtaining a printed ballot for voting, the voter interacts with an EBM and possibly a device that checks and verifies the WBB signatures. Potential failures and possible recovery measures are discussed below. There are several different potential failures, but they may be difficult to distinguish and hence have similar responses.

5.4.1. Demonstrable EBM malfunctions

The preference receipt is invalid For example, it is blank or it contains repeated numbers. This is an EBM error.

EBM receipt is printed but does not contain a valid digital signature: This is also an EBM error. An invalid or absent signature may be the result of simple device malfunction, but it may also have more serious implications such as failure to upload the vote to the WBB, or incorrect recording on the WBB.

Recommended action in both cases:
- The vote should be cancelled, then

- The voter should be issued with a fresh ballot and allowed to cast another vote.

- The EBM should be removed from use until the problem is resolved.

These steps should be auditable, *i.e.* it should be possible for voters and third parties to check cancelled votes on the WBB. See Section 4.4 for details of procedures.

5.4.2. Apparent malfunctions that may be due to voter, EBM, or private WBB error

In all of these cases, it is possible that the EBM malfunctioned but also possible that a voter did not follow the instructions correctly, or claimed that the EBM malfunctioned when it did not.

The EBM does not print a preference receipt The elector cannot know if the VEC captured their vote as they intended it because they cannot verify it.

The preferences receipt is different to what the elector expects Note that this is indistinguishable from an elector who is mistaken, frustrated or dissatisfied claiming that their receipt is different when in fact the device functioned correctly.

Recommended action in both cases:

- The vote should be cancelled, then

- The voter should be issued with a fresh ballot and allowed to cast another vote.

It is not possible to tell whether the failure was due to the EBM failing to send the correct vote, or a network failure, or the private WBB failing to sign and return what it received. Vote cancellation and re-voting are the most appropriate actions as the system may have failed to record the vote. However, it would be premature to remove the EBM from service following only a small number of accusations of this sort of behaviour. The authority may decide to treat sporadic, or a small number of errors as simple anomalies or voter mistakes. However, the authority should investigate the cause of a large number of reported errors.

5.5. Potential failures in post-election checking

Vote is incorrectly recorded on the WBB or absent from WBB: On discovering such an error, a voter can lodge a complaint. The first step would be for an official to check the reported error and the WBB signature on the receipt. If the vote is indeed wrongly recorded, and the voter has a validly signed receipt, this represents a serious system failure which implies that more than the assumed maximum number of private WBB peers have been compromised.

A voter who makes this complaint without a validly signed receipt does not have a strong case.

Final tally proof does not verify: An error found post-publication is more likely to be caused by a technical problem e.g., in uploading, than in the actual calculation. Clearly, the error would have to be investigated, the problem located and a solution sought. While it is potentially very serious, possibly indicating failure in the "back-end" processing, it could also be the result of a minor error that is easily fixed by, for example, uploading missing data.

If no simple error is found, it should be possible to identify the mix servers whose proofs were not valid (or not present) and rerun the mix without them, eventually producing a valid, verifiable proof.

6. Security Claims and analysis

Security requirements for voting systems fit into two main categories: integrity properties and privacy properties. We give a brief overview here and then more detail on each set of properties.

The protocol makes no trust assumptions for integrity for typical voters, apart from trusting that each eligible voter is allowed to cast at most one vote, and that only eligible voters can vote. It does of course rely on voters to perform some checks (an auditing assumption), which are detailed in Section 3. Invalid ballots, in which the candidate list does not match the encrypted ciphertexts on the WBB, are detected at ballot confirmation by Check 1. Check 2 detects incorrect vote printing by the EBM. Incorrect vote submission by the EBM before submission to the private WBB is detected by Check 3. Check 4 detects vote substitution by the WBB. Incorrect mixing or decryption would be detected because the proofs of correct mixing and decryption are public.

The vision-impaired voter is unable to do Check 2, that the EBM printed the correct ballot. She cannot ask for human assistance without destroying privacy. This leads to a distribution of trust over the machines in the polling place: she can check her vote on as many machines as she likes, and must assume that at least one of the machines she uses is honest.

The protocol not only detects, but also provides evidence of many kinds of failures, including the failure of ballot generation confirmations and the failure of a properly produced receipt to appear on the bulletin board. In both cases, a voter who experiences such a failure has a WBB digital signature that proves that the system malfunctioned. This provides two kinds of benefits: voters can demonstrate to a court that a real malfunction occurred, but it is not feasible to pretend that a malfunction occured when it did not. This is important in defending against the "defaming attack" in which people pretend to have detected a system failure which did not actually happen. Of course, there can be no proof that the EBM accurately represented the voter's intention: that step is dependent on the voter's testimony and hence is to an extent vulnerable to the "defaming" attack.

Privacy of the contents of each receipt depends on the assumption that at least two $RGen$ Server generate randomness correctly and keep it secret. Further, that a threshold set of those who share the keys is honest. If these assumptions hold, then the receipt itself does not leak information about the voter's preferred candidates (though it does show how many preferences they listed, and whether they voted ATL or BTL).

Provided that that two assumptions holds, the system has some defence against kleptographic attacks on the receipt [GKK+06]. This is because the receipt's random data is generated in a distributed way, and the entities that do the printing (the printer and the EBM) are deterministic. Thus information cannot be leaked in the ballot data itself

without some chance of detection, though it could be subtly leaked in slight font changes or other printing effects.

Privacy of the votes also depends on the privacy of the mixing protocol. If the mix is secure, then the tallying protocol does not add any information about the link between a receipt and its vote. RPC challenges are constructed so that overall anonymity across the mix is preserved.

The system is also receipt-free, meaning that it does not provide a person with information to prove how she voted. However, there are coercion attacks on this protocol, including the "Italian attack." These, along with other important details, are described below. We concentrate first on integrity properties and then discuss the subtleties of privacy.

This protocol is meant to achieve two main classes of security properties: integrity and privacy. The claims are given here and their informal justification is given below.

6.1. Integrity properties

vote integrity: meaning that all attempts to manipulate the votes are detectable by confirmation checks or other audits.[1]

non-repudiation: meaning that failures can not only be detected, but (in most cases) demonstrated. In particular, failures of the private WBB to post something it has accepted on the public WBB can be proven by producing the signed accepted item (whether a signed ballot confirmation or a signed receipt for a submitted vote). This also defends the system against people falsely claiming to have detected an error.

prevention of ballot stuffing: meaning that (under a threshold assumption, and a procedural assumption for voter markoff) only votes entered via the legitimate interface are included in the count.

Defences against ballot stuffing using the legitimate interface (e.g. by unauthorised people gaining access to a legitimate polling place) are not part of the system and must be defended against by procedural mechanisms.

Procedures must prevent voters from taking someone else's ballot off the printer and hence voting in the wrong division.

6.1.1. Justification of Integrity claims

Vote integrity based on confirmations

An informal argument for the integrity of each person's vote is:

---

[1] Of course this does not imply that they will always be detected, if the appropriate checks are not performed on the manipulated ballot. The claim is that any manipulation can in principle be detected if a check is performed.

- The ballot-generation confirmation checks that the ballot is a permutation of properly-encrypted candidate identifiers.

- The ballot-printing confirmation checks that the printed list of candidate names matches the encrypted candidate identifiers on the WBB.

- The voter's check of the EBM's printout confirms that the correct numbers (or other marks) are recorded against the correct candidate names.

- The signature check confirms that the printed number sequence matches what was submitted to the WBB.

- The check of the vote on the WBB confirms that the correct ciphertexts were used (in the case of a larger-than-necessary generated ballot) and that the vote submitted to the WBB was posted.

- verifying the shuffling and decryption proof from the mixnet confirms that the announced output votes match those posted (encrypted) on the WBB.

Of course the first two confirmations are performed only on ballots that are *not* subsequently voted on. The argument is that any attempt to manipulate the vote by generating or printing invalid votes will be detected by with some probability that depends on the confirmations being numerous and unpredictable.

6.1.2. Selecting Ballots for Confirmations and Audits

Clearly it is important that we use a suitable source of randomness for the selection of the ballots to do the confirmation checks. Some combination of public confirming (with officials, scrutineers, observers, and a public source of randomness such as dice or lotto balls) with voter-initiated confirmation checks (in which any voter may choose to confirm ballot construction or printing) would be ideal. This will require further investigation to see what procedures are possible in practice. The argument about the integrity of the results of the election depends on all steps of the confirming process having been performed diligently, including those that the voters have to do themselves.

Of course, it is difficult to compute the appropriate amount of confirming for an IRV/STV election, especially in advance [MRSW11, Car11]. This question will have to be addressed for the project, but is out of scope for this paper. We expect that most of the IRV (*i.e.* single-seat) contests will have a relatively easy margin computation in practice. However, the full STV contests are a different matter and require significant further thought. One possibility is to announce the result, explain what quantity of cheating might have been possible given reasonable estimates of the amount of confirming that was done, and ask any election challenger to demonstrate a set of votes in which they win a seat and the number of changed votes is reasonably probable given the rate of checking.[2]

---

[2]Thanks to Ron Rivest for this suggestion.

A Bayesian method of bounding the total number of errors for a particular confidence level, given a particular quantity of confirming, is given in Appendix A.

6.1.3. Properties of different kinds of mixnet

This design is largely independent of the kind of mixnet chosen to shuffle and decrypt the votes. However, it inherits the privacy properties of the mixnet it uses. The current implementation uses Randomised Partial Checking [JJR02], adapted according to recommendations by Khazaei and Wikström [KW13]. As described in Section 4.5, RPC mixnets allow a small but non-negligible probability of successful cheating, which decreases exponentially with the number of substituted votes.

Our implementation is designed to allow cheating only if the entire set of mixers collude to cheat, because they all work together to compute their challenges. Even if they all do, a given probability of detection requires an amount of precomputation work that is exponential in the number of substituted votes.

We intend to continue to evaluate whether a mixnet based on zero knowledge proofs such as Verificatum [Wik12] will be feasible to use in future.

6.1.4. Other integrity properties

Serial Number uniqueness and defence against the "clash attack"

The "clash attack" [KTV12] is a vote dropping technique that applies to many cryptographic voting schemes. An attacker (as a server or ballot generator) arranges to give several different voters identical receipts. All affected voters see their receipt appear on the public WBB, and yet only one vote has been counted. In our protocol, the serial numbers are carefully generated to guarantee their uniqueness (See Section 4.2.2), but this doesn't prevent a corrupt printer from printing off exactly the same ballot, with the same Serial Number, for many different voters. The printer would have to collude with a corrupt EBM that merely reused the private WBB signature, without resubmitting multiple instances of the same vote to the WBB. There are two reasons that this represents an acceptable risk.

- The attack works only if the voters subsequently cast identical votes—otherwise the cheating EBM will be unable to produce a valid signature on the receipt, and unable to post it to the WBB. (The attack is in general harder for Prêt à Voter than for direct-encrypting schemes such as Helios and Wombat, because the attacker must commit to the identical ballot before learning the person's vote.)

- The attack is detectable by ballot printing audit, which would fail because the ballot has already been voted on.

Overall this attack is no more effective, requires more conspirators, and has a higher probability of detection than the simple misalignment of the candidate names on the ballot by a corrupt printer. Hence it is appropriate (*i.e.* conservative) to include this attack implicitly in computations quantifying the extent of ballot misprinting, without explicitly counting it.

Non-repudiation and defence against the "defaming attack"

Every legitimate receipt includes a WBB signature on the SerialNumber, preferences and division. Voters are encouraged to check the signature before leaving the polling place. If a voter can produce such a receipt without it appearing on the WBB, then this demonstrates that the Private WBB has malfunctioned. Conversely, accusations that a particular receipt was properly submitted remain unconvincing when the claimed receipt does not have a valid WBB signature.

Ballot generation and printing confirmations can demonstrably fail, or demonstrably succeed, or fail silently (*e.g.* when the device being confirmed simply stops). In other words, if a voter claims to have received an invalid opening of a printed ballot, then that should be demonstrable because a printed ballot should have a valid WBB signature. However, if a voter claims to have attempted to confirm a ballot but not received a result, this cannot be checked.

Misprinting of voter intention by the EBM cannot be demonstrated, because only the voter knows what they truly entered. Consequently, accusations of incorrect printing cannot be repudiated—the evidence that a particular machine is misbehaving needs to consist of a series of observations and comparisons with other machines.

Prevention of ballot stuffing

The private WBB enforces that votes may only be uploaded to the WBB by a legitimate EBM casting a vote that has been properly printed and signed. In other words, a colluding printer and EBM could stuff the ballot, but this would be detected by the reconciling of markoff data with the number of submitted ballots described in Section 3. Also (a threshold of peers of) the private WBB could stuff the ballot, by pretending to have received a legitimate signed vote, but again this is detected by reconciling with markoff data.

It is not possible to include the EBM signature on the public WBB because the protocol does not guarantee that the same signature has been sent to all private WBB peers.

6.2. Privacy properties

privacy: meaning that the system hides how each person voted, under assumptions stated below.

It does reveal whether the person voted above or below the line, and (in the BTL case) how many preferences they expressed.

receipt freeness: meaning that even a voter who deliberately colludes with a coercer cannot prove after voting how they voted. Note that this implies that a voted ballot cannot also be confirmed.

resistance to kleptographic attacks: meaning that a printer attempting to leak information via the WBB data will be detected with some probability.

It is not intended to defend against "Italian attacks" or randomisation attacks (in which a voter is coerced to produce a receipt of a particular form, which has a random effect on the actual vote). Defences against coercion associated with failing to shred the candidate list must be enforced by procedural mechanisms. Similarly, deliberate uses of out-of-band recording technology (such as taking a photo of the candidate list before shredding it) or side-channel information leakage (from the EBM or printer) must be defended against by mechanisms outside the vVote system.

vVote also contains some technical and procedural measures, which are unrelated to Prêt à Voter, for defending against chain voting. These are described in Section 3.1.5.

6.2.1. Justification of privacy claims

The following coalitions can violate ballot privacy for an individual:

- The printer that generated and produced that ballot,

- All but one of the *RGen* servers,

- The EBM the voter used,

- However many mix servers are necessary for breaking shuffling privacy, depending on the mixnet being used,

- A threshold of key sharing authorities,

The crucial claim is that smaller coalitions cannot. This is expanded into several specific claims below.

Clearly if the printer leaks its information it can violate vote privacy for everyone who used a ballot it printed. This means that practical opportunities for compromising the printer must be reduced as much as possible, *e.g.* turning off the wireless connection.

Apart from the printer and an electronic ballot marker (if there is one), no other single entity can violate vote privacy. This is justified in two claims below.

Claim 1 *A collusion of all but two randomness generation authorities does not have sufficient information to recover the ballot permutation (in polynomial time with non-negligible probability).*

Clearly if all the randomness generation authorities collude and share their information, they learn the contents of all ballots. If at least two choose their random values correctly and keep them secret until the others have committed, and if the others can be forced to open their commitments, the resulting list of random values has $2k = 2 * 256$ bits of entropy.

NIST [BK12] states that the SHA family of hashes are suitable as randomness extractors. In [BK12] it states that when using a hash function F in which $Y = F(S||A)$ then "If the input string S was assessed at $2n$ bits of min-entropy or more (i.e., $m \geq 2n$), then Y may be considered to have n bits of full entropy output". The value of A can be

anything, including null, it is just additional data. This tells us that provided at least two mix servers provide good randomness values the output from the hashes will have $k = 256$ bits of entropy.

The usual subtlety arises if we consider the possibility that some authorities might use blocking to bias the output, *i.e.* might wait until learning the other authorities' random values and refuse to open their own commitments if they did not like the result. (This could happen, for instance, in collusion with a corrupt printer.) This is why true coin-tossing protocols are more complicated than the simple one in this proposal. In practice, such a blocking authority would be removed quickly without having the opportunity to affect many bits of the output.

Clearly the same argument holds for the PRNG construction of Section 4.2.2, given appropriate assumptions about the PRNG.

Claim 2 *The posted ballots on the WBB reveal, for each receipt, whether the vote was ATL or BTL and how many preferences were cast, but no other information, unless a threshold of key sharers colludes.*

Claim 3 *The mixing process anonymises votes within anonymity sets defined by their division, ATL/BTL choice, and length of preference list. The assumptions about mixnet collusion for privacy violations depend on the mixnet being used.*

For example, in RPC, if all but one pair of mix servers exposes their permutation, then each anonymity set is half the total being mixed.

In a mixnet based on zero knowledge proofs, votes are anonymised within the whole set if at least one mix server remains honest.

Our system is not susceptible to the replay attack described in [KW13] because all ballots are pregenerated in a distributed fashion.

6.2.2. Kleptographic attacks

The output of the printers is entirely determined by the randomness that is sent to them, and other publicly committed information given in Figures 4.1 and 4.2. Hence they have no opportunity to provide any information which may be skewed in a particular way. Correct information posted therefore cannot leak information from the printer. Incorrect information will be detected with some non-negligible probability by the ballot-generation confirmation processes.

Although the whole group of randomness generation authorities can collude to mount a kleptographic attack, a similar argument to that for vote privacy shows that a smaller collusion has insufficient information.

6.2.3. Receipt freeness

Receipt freeness is a subtle property and we do not have a formal argument for it. However, the main idea behind Prêt à Voter is to provide the voter with either a proof of the contents of a ballot's encrypted values, or an opportunity to vote on the ballot,

but never both for the same ballot. In other words, a ballot that's allowed to be voted on should never have revealed the random values used to produce it.

The threat of using the confirmation process to expose the contents of a ballot that has been voted on is ameliorated by the electronic locking process described in Sec 4.2.3.

A voter could attempt to collude with a corrupt printer to produce a receipt, and could promise not to perform a ballot printing confirmation check, but the incorrect formation of the ballot necessary to produce such a receipt would be caught by a ballot construction confirmation check with some probability.

6.2.4. Privacy Threats Ameliorated By Procedural Controls

As the voter inputs her choices into the EBM, the device necessarily "learns" how she voted. The potential for the EBM to leak vote information clearly raises privacy issues. Any data stored in the EBM's memory should be deleted, ideally after each session.

Prêt à Voter introduces a privacy threat that does not exist for either standard paper voting or for DRE's with VVPAT: someone may discover and record an unvoted ballot's candidate order and look up code, then learn the vote choices when they are later posted on the WBB. Therefore there should be procedural controls to protect both the paper printout and the electronic data on the printer from observation by anyone but the voter.

As for any voting system, computerised or paper-based, voters may ask for assistance at a point that potentially violates their privacy simply because the assistant sees what the voter has already written or entered. This threat to privacy however, exists in the current system.

6.2.5. Privacy issues arising from small populations and complex ballots

Sometimes vote privacy is unachieveable: if everyone in one polling place votes the same way, and results for that polling place are observed or announced alone, then vote privacy is not possible. It is possible that some divisions will have very few vVote voters.

The small populations exacerbate existing problems caused by the complexity of Victorian ballots. It is easy for a voter (or a coercer) to choose a BTL vote that is highly likely to be unique. If the BTL votes are made public, this allows a voter to prove how they voted.

Since vVote receipts expose whether the person voted ATL or BTL, and how many preferences they expressed, this problem is exacerbated again: it may be possible to identify a vote uniquely based on its division and number of BTL preferences. If only one person cast a particular number of preferences below the line in a division, that person's vote would have to be withheld from publicly verifiable decryption.

Some techniques exist for ameliorating some of these problems, such as proposals for verifiable privacy-preserving tallying using (more) mixing and homomorphic sums [BMN$^+$09, Hea07]. However, these may not be efficient enough to use in practice. More importantly, they work only on a complete list of votes, while the vVote votes need to be input into an (unencrypted) existing VEC counting system.

This leaves us with some ad hoc approaches that are highly dependent on how many people vote in each division, how close the election result is, and what anonymity thresholds are deemed acceptable. The problem could be mitigated by doing on-demand decryption of the packed candidate IDs, so we only decrypt the next pack when the previous one has been eliminated. However, that still does not guarantee that privacy will be preserved, since if all preferences are counted the full information will be made public anyway.

6.2.6. Other possible attacks

"Psychological" attacks are a potential threat. As an example, a coercer manages to convince voters that he is able to decrypt their receipts and find out how they voted [RP05]. Voter education could mitigate this attack; however psychological attacks will be a problem for virtually any end-to-end verifiable system.

Acknowledgements

References

[AN06]   Ben Adida and C Andrew Neff. Ballot casting assurance. In *Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop*. USENIX Association, 2006.

[BBB±13]   Susan Bell, Josh Benaloh, Michael D. Byrne, Dana DeBeauvoir, Bryce Eakin, Gail Fisher, Philip Kortum, Neal McBurnett, Julian Montoya, Michelle Parker, Olivier Pereira, Philip B. Stark, Dan S. Wallach, and Michael Winn. STAR-vote: A secure, transparent, auditable, and reliable voting star-vote: A secure, transparent, auditable, and reliable voting system. *USENIX Journal of Election Technology and Systems (JETS)*, 1(1), August 2013.

[BCH+12a] Craig Burton, Chris Culnane, James Heather, Thea Peacock, Peter Y. A. Ryan, Steve Schneider, Sriramkrishnan Srinivasan, and Zhe Xia. A supervised verifiable voting protocol for the Victorian Electoral Commission. In *Proc. 5th International Conference on Electronic Voting*, 2012.

[BCH±12b] Craig Burton, Chris Culnane, James Heather, Thea Peacock, Peter YA Ryan, Steve Schneider, Sriramkrishnan Srinivasan, Vanessa Teague, Roland Wen, and Zhe Xia. Using Prêt à Voter in Victorian State elections. In *Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*, 2012.

[Ben06]   Josh Benaloh. Simple verifiable elections. In *Proc. 1st USENIX Accurate Electronic Voting Technology Workshop*, 2006.

[BK12]   Elaine Barker and John Kelsey. NIST DRAFT Special Publication 800-90b recommendation for the entropy sources used for random bit generation. Technical report, NIST, 2012.

[BL11]   Josh Benaloh and Eric Lazarus. The trash attack: An attack on verifiable voting systems and a simple mitigation. Technical Report MSR-TR-2011-115, Microsoft, 2011.

[BLS01]   Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In *ASIACRYPT 2001*, pages 514–532. Springer, 2001.

[BMN±09]   Josh Benaloh, Tal Moran, Lee Naish, Kim Ramchen, and Vanessa Teague. Shuffle-sum: coercion-resistant verifiable tallying for STV voting. *IEEE Transactions on Information Forensics and Security*, 4(4):685–698, 2009.

[BNFL+12] Jonathan Ben-Nun, Niko Fahri, Morgan Llewellyn, Ben Riva, Alon Rosen, Amnon Ta-Shma, and Douglas Wikström. A new implementation of a dual (paper and cryptographic) voting system. In *5th International Conference on Electronic Voting (EVOTE)*, 2012.

[Car11] David Cary. Estimating the margin of victory for instant-runoff voting. In *USENIX Accurate Electronic Voting Technology Workshop Workshop on Trustworthy Elections*, 2011.

[CCC+10] Richard Carback, David Chaum, Jeremy Clark, John Conway, Aleksander Essex, Paul S. Herrnson, Travis Mayberry, Stefan Popoveniuc, Ronald L. Rivest, Emily Shen, Alan T. Sherman, and Poorvi L. Vora. Scantegrity II municipal election at Takoma Park: The first E2E binding governmental election with ballot privacy. In *Proc. USENIX Security*, 2010.

[CHJ±13] Chris Culnane, James Heather, Rui Joaquim, Peter Y. A. Ryan, Steve Schneider, and Vanessa Teague. Faster print on demand for Prêt à Voter. *USENIX Journal of Election Technology and Systems*, 2(1), 2013.

[CHPV09] David Chaum, Benjamin Hosp, Stefan Popoveniuc, and Poorvi L. Vora. Accessible voter-verifiability. *Cryptologia*, 33(3):283–291, 2009.

[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.

[CS14] Chris Culnane and Steve Schneider. A peered bulletin board for robust use in verifiable voting systems, 2014. arXiv:1401.4151.

[DC07] Roberto Di Cosmo. On privacy and anonymity in electronic and non electronic voting: the ballot-as-signature attack, 2007.

[FS01] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In *CRYPTO 2001*, pages 368–387. Springer, 2001.

[GKK+06] Marcin Gogolewski, Marek Klonowski, Przemyslaw Kubiak, Miroslaw Kutylowski, Anna Lauks, and Filip Zagórski. Kleptographic attacks on e-voting schemes. In *International Conference on Emerging trends in Information and Communication Security*, pages 494–508, 2006.

[GTT11] Michael T. Goodrich, Roberto Tamassia, and Nikos Triandopoulos. Efficient authenticated data structures for graph connectivity and geometric search problems. *Algorithmica*, 60(3):505–552, 2011.

[Hea07] James Heather. Implementing STV securely in Prêt à Voter, 2007. Computer Security Foundations.

[JJR02]   Markus Jakobsson, Ari Juels, and Ronald Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX Security Symposium*, pages 339–353, 2002.

[KTV12]   R. Küsters, T. Truderung, and A. Vogt. Clash Attacks on the Verifiability of E-Voting Systems. In *IEEE Symposium on Security and Privacy (S&P 2012)*, pages 395–409. IEEE Computer Society, 2012.

[KW13]   Shahram Khazaei and Douglas Wikström. Randomized partial checking revisited. In *Topics in Cryptology, CT-RSA 2013*, pages 115–128. Springer, 2013.

[MRSW11] Thomas R. Magrino, Ronald L. Rivest, Emily Shen, and David Wagner. Computing the margin of victory in IRV elections. In *USENIX Accurate Electronic Voting Technology Workshop Workshop on Trustworthy Elections*, 2011.

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

[RBH$^{\pm}$09] Peter Y. A. Ryan, David Bismark, James Heather, Steve Schneider, and Zhe Xia. Prêt à voter: a voter-verifiable voting system. *IEEE Transactions on Information Forensics and Security*, 4(4):662–673, 2009.

[RP05]   Peter Y.A. Ryan and Thea Peacock. Prêt à Voter: a systems perspective. Technical Report CS-TR-929, University of Newcastle upon Tyne, 2005.

[RS06]   Peter Y.A. Ryan and Steve Schneider. Prêt à Voter with Re-encryption Mixes. In *European Symposium on Research in Computer Security*, number 4189 in Lecture Notes in Computer Science. Springer-Verlag, 2006.

[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 *Proc. 17th USENIX Security Symposium*, 2008.

[Sta09]   Philip B. Stark. Risk-limiting post-election audits: *P*-values from common probability inequalities. *IEEE Transactions on Information Forensics and Security*, 4:1005–1014, 2009.

[TW10]   Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In *Progress in Cryptology–AFRICACRYPT 2010*, pages 100–113. Springer, 2010.

[Vic07]   Victorian Electoral Commission. Report to Parliament on the 2006 Victorian State election, July 2007.

[Wik12]   Douglas   Wikström.   Verificatum,   2012.
            http://www.verificatum.org/verificatum/.

[XCH±10]   Zhe Xia, Chris Culnane, James Heather, Hugo Jonker, Peter Y. A. Ryan,
            Steve A. Schneider, and Sriramkrishnan Srinivasan. Versatile Prêt à Voter:
            Handling multiple election methods with a unified interface. In *INDO-CRYPT*, pages 98–114, 2010.

A. Audit Sample Sizes and Confidence Levels: a Bayesian Analysis

In vVote there are several stages of processing the ballots and votes which are checked by sampling: ballot generation, ballot printing, printing of preferences, posting the preferences on the web bulletin board, and mixing by the re-encryption mixnet. This section considers the rate of sampling necessary at each of these stages in order to make statements about the probability of there having been an undetected attack altering sufficient ballots to change the result.

Stages are checked by sampling: selecting some of the ballots/votes being processed, and checking that they have been processed correctly. If a ballot passes a check then it has not been altered. The stages to check are as follows:

Ballot generation: Cut and choose: ballots are sampled and checked that the candidate identities have been encrypted correctly. Sampled ballots are not used for voting, since the confirmation check would break secrecy of the ballot. This check is carried out by the authorities during the ballot generation phase.

Ballot printing: Cut and choose: printed ballots can be challenged by a voter, who can have it checked that the printed candidate list has been correctly printed. Checked printed ballots are not used for voting since secrecy would be lost. This check is carried out by the individual voters.

Printing of preferences: Voters check that the preferences on their signed printed receipt are in the correct order. Carried out by individual voters.

Checking the WBB: Voters check that the entry on the WBB matches their receipt. Carried out by individual voters.

Mixnet: Randomised partial checking of the links in the mixnet. Carried out by the authorities.

We are interested in the claims we can make when all of these checks are successful.

The kind of claim we wish to make is that **given a sampling rate of r and no failed checks, the probability that at least F votes were altered is no more than** p. Typically F will be some small fraction of the total number of votes cast. The value F of interest may be the smallest number of votes necessary to change the result.

The converse is straightforward to calculate: if F votes are altered, then the probability that this is not detected by sampling at rate r will be $(1 - r)^F$ (see Appendix A.2). We can then use Bayes' theorem to obtain the probability that there is an attack given that the sampling does not find any cheating, as follows:

Given a prior (pre-sampling) probability q that there has been an attack affecting F ballots, we can calculate the probability p that there has been an attack affecting F ballots given the sampling. This is given by the following Bayesian formula, which is explained more fully in Section A.3:

$$p \;=\; \frac{(1-r)^F.\, q}{(1-r)^F.\, q + (1-q)} \tag{A.1}$$

A.1. Digression: combining confirmations of different stages

Given stages 1 to n, to change a total of F votes an adversary may change some votes at each stage, to a total of F. We consider F_i votes changed at stage i, where $F_1 + \ldots + F_n = F$.

Each stage may have a different rate of sampling. At stage i we consider the rate of sampling to be r_i. The probability that the sampling at stage i will not find any of the F_i altered ballots is $(1-r_i)^{F_i}$. Hence the overall probability p_0 that none of the attacked ballots will be discovered will be $p_0 = (1-r_1)^{F_1}.(1-r_2)^{F_2}.\ldots.(1-r_n)^{F_n}$.

Let r_J be the least of all the rates. Then (replacing each r_i by r_J), p is bounded above by $(1-r_J)^F$. Hence we have the highest probability that none of the attacked ballots will be discovered when we make all the changes in the stage with the lowest sampling rate.

Hence when we identify a sampling rate required to achieve particular assurances, we require that the identified rate will be the *minimum* that any of the stages have. This means that in *all* the stages of ballot processing which involve auditing, the sampling must be at the level of *at least* at the required rate.

Note that the RPC check of the mixnet requires a different argument because it is links in the mix rather than individual ballots that are checked. In practice the best known attack on mixnet integrity allows a vote to be altered with a 25% probability of detection, and can thus be treated as a 25% rate of sampling. In practice this will be higher than the sampling in the other stages.

F=100 Sampling Rate	a priori probability of cheating					
	1%	2%	5%	10%	20%	50%
1.0%	0.4%	0.7%	1.9%	3.9%	8.4%	26.8%
2.0%	0.1%	0.3%	0.7%	1.5%	3.2%	11.7%
3.0%	0.0%	0.1%	0.2%	0.5%	1.2%	4.5%
4.0%	0.0%	0.0%	0.1%	0.2%	0.4%	1.7%
6.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.2%

Figure A.1.: Calculations of probability of fraud given no fraudulent ballots in sample

A.1.1. Example

Figure A.1 gives an example table, considering the probability for $F = 100$ of fraudulent ballots given a percentage sample size which finds no fraudulent ballots, and for various a priori probabilities of attack.

For example, suppose that we have an a priori expectation from our other information (e.g. no other circumstantial evidence of any attack behaviour) that there is at most a 5% chance of an attack, so $q = 0.05$. Suppose also that $r = 3\%$ and that $F = 100$. Then we have $(1 - r)^F = 0.05$, and we obtain $p = 0.002$.

In other words, if we sample 3% of the ballots, and we find no fraudulent ones, and we also estimate that there is no more than a 5% probability of an attack, then the probability of there being an attack on 100 ballots given the sampling is $0.002 = 0.2\%$.

If the prior probability of an attack was at the higher value of 10% then after the sampling the probability of there having been an attack will be 0.5%.

A.1.2. Required sampling rates

In practice we will start with a number F of fraudulent ballots that we want to make a claim about; a judgement on the prior probability q of an attack; and a confidence level $1 - p$ that we want a sample check to give us. Then we can calculate the rate of sampling that we need. This is given in Figure A.2 for confidence level 99.5%, and in Figure A.3 for confidence level 95%.

We can see from the chart how the level of sampling required changes with the prior probability q of cheating, particularly as F increases. For example we can see that if there is a 10% prior probability of cheating, and we are concerned with whether $F = 100$ ballots have been changed, then we need to sample 3.0% of the ballots, and find no cheating, to be able to conclude with 99.5% confidence that this level of cheating has not occurred: in other words, the probability that there was no attack is 99.5% given that this level of sampling finds no cheating. However, we see that even with an extremely high prior probability of cheating of 50%, we still only need to sample 5.2% of the ballots to achieve the same level of confidence that cheating has not occurred.

For a 95% confidence level of no cheating on $F = 100$ and a prior probability of 10% cheating, Figure A.3 shows that we only need to sample 0.7% of the votes.

Conversely, Figure A.4 is appropriate for post-hoc analysis: given a value of F, and an observed level of sampling, the table gives the probability of there having been cheating to the level of F altered ballots for a prior probability of 5%.

Note that it is the voters themselves who decide to check the printing of the ballots, and who decide whether to confirm their votes on the WBB, so the rate of confirmation checks for ballot printing cannot be decided in advance. However, mitigations such as additional print confirmation checks can be carried out by "mystery shoppers" to achieve the required rate.

Reconsider the Bentleigh 2010 example with a winning margin of 522 votes. Following our analysis, if there is no evidence of any attack then we might reasonably bound the prior probability of cheating at 10%, (though this figure would require some careful

F	a priori probability of cheating						
	1%	2%	5%	10%	20%	50%	99%
10	6.7%	13.1%	20.9%	26.6%	32.3%	41.1%	62.8%
50	1.4%	2.8%	4.6%	6.0%	7.5%	10.0%	17.9%
100	0.7%	1.4%	2.3%	3.0%	3.8%	5.2%	9.4%
250	0.3%	0.6%	0.9%	1.2%	1.6%	2.1%	3.9%
500	0.1%	0.3%	0.5%	0.6%	0.8%	1.1%	2.0%

Figure A.2.: Sampling rates required to achieve **99.5%** probability that that there was no cheating to the level of F altered ballots.

F	a priori probability of cheating					
	6%	10%	20%	30%	50%	99%
10	1.9%	7.2%	14.4%	18.9%	25.5%	52.9%
50	0.4%	1.5%	3.1%	4.1%	5.7%	14.0%
100	0.2%	0.7%	1.5%	2.1%	2.9%	7.3%
250	0.1%	0.3%	0.6%	0.8%	1.2%	3.0%
500	0.0%	0.1%	0.3%	0.4%	0.6%	1.5%

Figure A.3.: Sampling rates required to achieve **95%** probability that that there was no cheating to the level of F altered ballots.

F	sampling rate				
	1%	1.5%	2%	3%	4%
50	3.1%	2.4%	1.9%	1.1%	0.7%
100	1.9%	1.1%	0.7%	0.2%	0.1%
200	0.7%	0.3%	0.1%	0.0%	0.0%
300	0.3%	0.1%	0.0%	0.0%	0.0%
400	0.1%	0.0%	0.0%	0.0%	0.0%

Figure A.4.: Probability that that there was no cheating to the level of F altered ballots for given sampling rates, with a prior probability of cheating of 5%.

consideration and seems at the high end). Figure A.3 shows us that for a 95% confidence level of no attack we need only sample 0.3% In fact a sampling rate of 1.2% gives a confidence level of 99.5%, as shown in Figure A.2.

A.2. Probability that a sample passes the check given F fraudulent ballots

Define:

N The total population size (total number of ballots)

S The sample size: the number of ballots we will check

F The number of fraudulent (incorrectly constructed) ballots we are concerned about.

r The sample rate: sample size as a proportion of the total population size: $r = S/N$.

We'll also assume that $F \ll N$: that a very small proportion of the total ballots are fraudulent.

If there are F fraudulent ballots out of N it the following calculation determines the probability that a sample size of S will not include any of the F fraudulent ballots:

$$p = \Pi_{i=0}^{S-1}((N - F - i)/N - i)$$

or

$$p = (\Pi_{i=0}^{S-1}(N - F - i))/(\Pi_{i=0}^{S-1}(N - i))$$

which simplifies to

$$\begin{aligned}
p &= (\Pi_{i=0}^{F-1}(N - S - i))/(\Pi_{i=0}^{F-1}(N - i)) \\
&= \Pi_{i=0}^{F-1}((N - S - i)/(N - i)) \\
&\approx ((N - S)/N)^F \qquad \text{when } F \ll N \\
&= (1 - r)^F
\end{aligned}$$

Hence we can obtain the probability that the sample passes the check (i.e., discovers no fraud) given F fraudulent ballots, will be approximately $(1 - r)^F$.

A table of example confidence levels for sample sizes against number of ballots corrupted is given in Figure A.5.

A table of sample rates to achieve particular confidence levels against number of ballots corrupted is given in Figure A.6. For example, to achieve a 99% confidence that there are fewer than 1000 incorrect ballots, it is necessary to check 0.5% of the total number of ballots.

percentage to check					
number cheating	1%	2%	5%	10%	25%
10	9.6%	18.3%	40.1%	65.1%	94.4%
20	18.2%	33.2%	64.2%	87.8%	99.6829%
50	39.5%	63.6%	92.31%	99.48%	99.9999%
100	63.40%	86.74%	99.41%	100.00%	100.0000%
250	91.89%	99.36%	100.00%	100.00%	100.0000%
500	99.343%	99.996%	100.0000%	100.0000%	100.0000%
1000	99.9957%	100.0000%	100.0000%	100.0000%	100.0000%

The cell gives the confidence level that there are not N incorrectly constructed ballots, given the percentage of the ballots that have been checked without finding any incorrect ones.

Figure A.5.: Confidence levels

Confidence level					
number cheating	95%	97.5%	99%	99.5%	99.9%
10	25.9%	30.8%	36.9%	41.1%	49.9%
20	13.9%	16.8%	20.6%	23.3%	29.2%
50	5.8%	7.1%	8.8%	10.1%	12.9%
100	3.0%	3.6%	4.5%	5.2%	6.7%
250	1.2%	1.5%	1.8%	2.1%	2.7%
500	0.6%	0.7%	0.9%	1.1%	1.4%
1000	0.3%	0.4%	0.5%	0.5%	0.7%

The cell gives the percentage of ballots to check (and confirm correct, with no incorrect ones) to achieve the confidence level that there are not N incorrectly constructed ballots

Figure A.6.: Percentage sampled to achieve confidence levels

A.3. Bayesian Analysis

Bayesian analysis is concerned with the way the probability of some occurrence A is affected by particular observations B. In our case we are interested in the probability that there are F fraudulent ballots in the case where sampling does not identify any fraud. Although we do not have a prior probability for fraudulent ballots, we we are still able to consider how sensitive the resulting probability is to possible prior probabilities.

The calculations above give the probability that the sample passes the check given there are F fraudulent ballots. Define:

A : there are F fraudulent ballots

B : the sample passes the check

$p(A)$: the probability of A

$p(\neg A)$: the probability of A being false

$p(A|B)$: the probability of A given B

Bayes' theorem states that

$$p(A|B) \quad = \quad \frac{p(B|A).p(A)}{p(B|A).p(A) + p(B|\neg A).p(\neg A)} \tag{A.2}$$

For the worst case analysis we will assume that there are either no cheating (0 fraudulent ballots) or that there are exactly F fraudulent ballots. In other words, if there is an attack then the attacker chooses exactly the right number to change the result.

We can therefore calculate $p = P(A|B)$, the probability we are interested, given any particular value of $P(A)$, the prior probability that there has been an attack which altered F ballots.

The formula in Line A.1 above is Line A.2 with the following substitutions:

- $p(B|\neg A) = 1$ — if no cheating then the probability that the sample passes the check must be 1.;

- $p(B|A) = (1 - r)^F$ — given earlier;

- $p(A) = q$ — prior (pre-sampling) probability of an attack. Must be estimated;

- $p(A|B) = p$ — the probability we are interested in.