MERGE: Matching Electronic Results with Genuine Evidence, for verifiable voting in person at remote locations

No Author Given

No Institute Given

Abstract

Overseas military personnel often face significant challenges in participating in elections due to the slow pace of traditional mail systems, which can result in ballots missing crucial deadlines. While internet-based voting offers a faster alternative, it introduces serious risks to the integrity and privacy of the voting process. We introduce the MERGE protocol to address these issues by combining the speed of electronic ballot delivery with the reliability of paper returns. This protocol allows voters to submit an electronic record of their vote quickly while simultaneously mailing a paper ballot for verification. The electronic record can be used for preliminary results, but the paper ballot is used in a Risk Limiting Audit (RLA) if received in time, ensuring the integrity of the election. This approach extends the time window for ballot arrival without undermining the security and accuracy of the vote count.

1 Introduction

Slow mail is one of the main barriers to electoral participation for many overseas military personnel. Some jurisdictions send blank ballots by paper mail and require ballot return by the same channel—this can be very slow, often resulting in ballots that are not returned by the deadline. Other jurisdictions allow for paperless voting by Internet, including email or pdf upload—this is very fast, but introduces unacceptable risks to integrity and privacy. One compromise is electronic delivery and paper returns, in which voters download a blank ballot and mail back a printed paper vote, having filled it in either manually or electronically. This has similar integrity and privacy properties to vote-by-mail, and takes only half the time that a fully paper solution would take. Nevertheless, the return mail delivery is often still too slow.

This paper describes the MERGE protocol, which enhances the basic scheme of electronic delivery and paper returns in a way that increases speed without significantly compromising privacy or integrity. The main idea is to send an untrusted electronic record back quickly, which can be confirmed if the paper ballot arrives in time for the Risk Limiting Audit (RLA), or distrusted otherwise. The RLA is a post-election auditing process that uses statistical methods to ensure with high confidence that the reported election outcome is correct, by manually comparing a random sample of paper ballots against the electronic vote tallies. Since RLA normally happens a few days to a few weeks after the election day, this extends the time available for the paper ballot to arrive.

Assume a public bulletin board (BB) which is an un-erasable, authenticated broadcast channel with memory. We will extensively use the assumption that people in different locations can rely on seeing the same data on the BB.

- In the polling place, the voter makes both a verifiable paper record, which is placed inside an envelope and mailed, and an untrusted electronic record, which is encrypted and posted to the BB.
- The voter is issued a digital signature, presented on a sticker, which serves as verifiable evidence of participation. Envelopes are considered acceptable only if they bear stickers with valid signatures upon receipt. This also allows officials at the Local Counting Center to notify the voter when their properly-signed envelope has arrived.[1]
- The untrusted electronic record can be incorporated into preliminary results and used as the cast vote record in an RLA.
- If the paper ballot arrives in time for the RLA, it is used as the ballot, just like any other RLA.
- If no paper ballot arrives that corresponds to a given electronic record, we use the phantoms-to-zombies approach of Bañuelos and Stark [2] to ensure that the risk limit is met even though the electronic record is not trusted.

Two assumptions are unavoidable: first, there must be an accurate count of people who legitimately participated; second, the paper ballot the voter places into the envelope must accurately reflect their intentions. Our protocol also requires someone to check the digital signature on every envelope. These all need to be supported by human verification processes.

The paper and electronic records could be produced by scanning a hand marked paper ballot, or printing a paper ballot from a ballot marking device (BMD).[2] See section 3 for a detailed discussion of the trust assumptions.

Compared to plain electronic delivery and paper returns, MERGE's main advantage is that, rather than needing ballot papers to arrive in time to be scanned for preliminary results, a vote can be safely counted if it arrives before the RLA. Of course, an electronic commitment to the vote must still be made before the normal close of polls. In practice, for most jurisdictions, this allows

[1] This is expected to be part of the practical implementation. It does not, however, form an important part of the cryptographic protocol because it does not include evidence that the paper ballot matched the electronic one. The protocol could be extended, with some extra mixing, to notify each voter of whether their specific ballot matched. The current version simply produces a collective tally of how many of the received ballots matched.

[2] Scientific opinion is divided on whether human verification of BMDs is sufficiently accurate to justify the assumption that the printout matches the voter's intention. See [1] and [10] for examples of opposing views. At a minimum, this requires careful design to ensure voters are motivated and encouraged to verify, and there is something they can do if a misprint occurs.

a few more weeks for mail to arrive. Its main disadvantage is that—if a large number of ballot papers do not arrive in time—the consequent large discrepancies may cause an RLA to fall back to a manual recount, when it would not have if those votes had simply been excluded from the beginning.

MERGE achieves *Software Independence* [11] based on a collective, probabilistic method of cast-as-intended verification designed to fit in to an existing Risk Limiting Audit. We assume that the attacker can control any electronic components, but at least one of the devices chosen for each verification step is honest. See subsection 1.3.

The election occurs in two classes of locations.

Remote Voting Centers are controlled polling places in remote areas such as overseas military bases, where voters are able to vote privately without coercion. There are clear instructions (e.g. on a poster on the wall) that do not come from the device used for voting, and some process for ensuring that the protocols assumptions are met.

Local Counting Centers are state- or county-based electoral authorities, where the ordinary votes from most citizens are counted. These will also receive paper ballots from Remote Voting Centers and process them.

There are likely to be many Remote Voting Centers and many Local Counting Centers, but we assume only one of each in this paper for simplicity.

1.1 Authentication using CAC cards

Voters at Remote Voting Centers own a smart card called a Common Access Card (CAC) card. It provides a Public Key Infrastructure (PKI) to enable secure authentication and digital signatures. Each CAC card includes a (private) signing key and a certificate linking its ID to the corresponding public key. Observers and officials at each Local Counting Center know the CAC IDs and certificates of the voters to be included. This knowledge is obtained either through individual CAC voters, who must register their public keys with their local electoral authorities, or via a more centrally organized process. Only votes with a valid CAC signature are accepted via the electronic channel.

Changes to CAC cards during the protocol, such as name changes, eligibility changes or key refreshes between the time the person votes and the time the votes are tallied, are out of scope for this paper but will need to be handled with specific policies in practice.

1.2 MERGE within the voting ecosystem

We assume that most votes are not from MERGE, but are cast locally near the Local Counting Center. The usual audit process is conducted there, with MERGE ballots incorporated from Remote Voting Centers as needed.

The ballot manifest—available to observers at the Local Counting Center—includes both the ordinary local ballots and all the ballots from Remote Voting Centers. So do the preliminary Cast Vote Records (CVRs).

The RLA proceeds exactly as it would if all the votes had been cast locally: observers see the ballot manifest and preliminary CVRs, then they watch a transparent process for seeding the PRNG that will be used to generate ballot samples, then they check the sequence of sampled ballots. For local ballots, CVRs are made in the usual way (e.g. by scanning); for MERGE ballots, preliminary CVRs come from the BB in encrypted form. If the sampled ballot is local, officials retrieve it in the usual way and compute its effect on the audit statistics. If the ballot was cast remotely, it will require some extra work to locate the mailed ballot and compare it with its electronic counterpart. It is also possible that the ballot was delayed in the mail—we deal conservatively with this situation. Figure 1 shows how MERGE fits into the voting and auditing process.

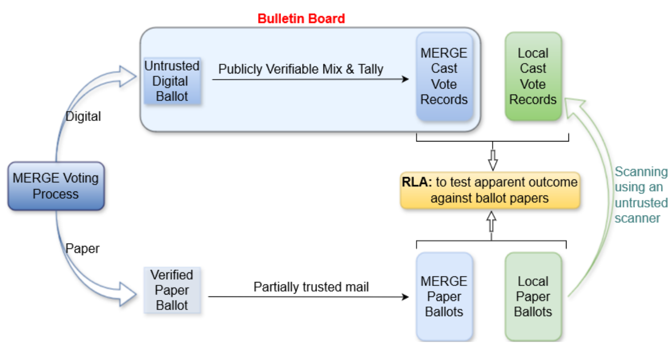

Fig. 1: Overview of MERGE in the voting ecosystem.

1.3 Security properties and contributions

- **Authentication:** Only eligible voters can vote.
- **Privacy:** The protocol does not reveal more about an individual's vote than the published tallies do. See section 5 and Appendix C.2.
- **Receipt Freeness:** It is infeasible for a voter to convince anyone of the value of their vote, even if they actively collude with the coercer, unless the coercer observes the serial numbers during the audit process. MERGE reveals who participated, so it is possible to coerce someone to refrain from participating altogether. See Appendix C.3.
- **(Individual) paper-based cast-as-intended verification:** Each voter can see that their paper vote accurately reflects their intention and refuse to cast it if it does not.
- **(Public) tally verification:** anyone can verify that the recorded electronic votes are correctly tallied.

– **(Collective, probabilistic) recorded-as-cast verification:** if the apparent election outcome (incorporating the electronic tally) does not accurately reflect the paper evidence (including both MERGE and ordinary ballots), the RLA will fall back to a full manual count except with probability at most the risk limit.

The formalization and proof of the last property is a major contribution of this paper: we employ a game-based approach to demonstrate that MERGE can interface with an existing RLA procedure, ensuring that the *overall* risk limit is still met. While [9] also uses a "physical cryptography game" framework to analyze RLAs in adversarial contexts, our work extends this approach to a complete e-voting protocol, addressing the complexities of processing both electronic and paper ballots prior to the RLA process. This property is different from (and neither weaker nor stronger than) end-to-end verifiability: although individuals cannot verify that *their* vote is properly recorded on the BB, the system can instead guarantee a global limit on the risk that enough votes are misrecorded to alter the outcome.

Guarantees achieved in this paper rely entirely on the trustworthiness of the paper trail, which is outside the control of the cryptographic protocol, and may fail for a variety of reasons, such as security problems in the mail channel and difficulties verifying printouts. These need to be addressed by human procedures outside the MERGE protocol.

1.4 Related work

There are many protocols for remote electronic voting, some with very strong security properties of various kinds. However, there is no single protocol that combines good privacy, receipt freeness (even against an attacker who can see only the bulletin board), public verifiability, and voter verification usable enough to constitute a good basis for trust. Since we work in a controlled setting, direct voter verification of a plaintext paper vote has substantial advantages, but the remote setting introduces practical challenges. This is a relatively under-studied area—most designs focus on either an uncontrolled Internet setting, or an in-person setting in the normal polling place.

Many systems provide end-to-end verifiability for polling-place voting and combine it with plain paper verification. Most could be adapted to remote, controlled polling places, but do not have detailed designs for dealing with ballots that are lost in the mail. An exception is ElectionGuard [3], which includes a vote-by-mail design, but this is intended for situations where preprinted ballots can be sent to voters and returned. It does not attempt to solve the timing issues associated with two-way mail. "Verifiable and Private vote-by-mail" [8] adds cryptographic verifiability to the electronic delivery and paper return strategy, much like MERGE. The main difference is that there is no pre-emptive electronic record sent, so no opportunity to take advantage of a timing difference between initial canvas and audit.

1.5 Implementation

Our open-source protoype implementation of MERGE uses elliptic curve exponential ElGamal encryption with threshold distributed decryption. It includes all of the protocol except the Bulletin Board. Details are in Appendix F.

2 Technical Background and Notation

2.1 Cryptographic components

ElectionGuard ElectionGuard [4] is a suite of open-source tools to support end-to-end verifiable voting. It uses a variant exponential form of the ElGamal cryptosystem to encrypt a vote. (See Appendix B.1 for details.) This allows:

- homomorphic addition of encrypted values, enabling the decryption of only the overall election tally,
- re-randomization of an encrypted value, which produces a ciphertext indistinguishable from a fresh one,
- threshold distributed decryption, which prevents any set of fewer than a threshold of talliers from colluding to decrypt individual votes.

Our design uses the ElectionGuard toolkit and relies on the paper record rather than trying to gain trust in the electronic record, with some extra details to allow for the sorts of failures that can happen when votes are mailed.

Homomorphic addition is written like multiplication on ciphertexts, either $\cdot$ for a pair or Π for a product of multiple values. The encryption of vote vector $\mathbf{b}$ is denoted by $\mathbf{e} = E(\mathbf{b})$.

Non-interactive zero-knowledge (NIZK) proofs The election transcript includes Noninteractive Zero Knowlege Proofs (NIZKPs) that allow for public verification that input votes are valid and that all decryptions are computed properly.

A NIZK proof of validity for an encrypted vote vector $\mathbf{e} = E(\mathbf{b})$ is denoted by $ZKP_{\text{VALID}}(\mathbf{b})$. It proves:

- the encryption for each option (e.g. e_j) encodes a valid value, usually either zero or one (i.e. $b_j \in \{0, 1\}$), and
- the sum of all encrypted options within each contest falls within a specific range (determined by the voting rules).

We also use ElectionGuard's NIZK proofs to demonstrate that an encrypted value (e.g. $e = E(b)$) is decrypted to a specific value (e.g. b), without revealing the decryption key. This proof is denoted by $ZKP_{\text{DEC}}(b, e)$.

Private decryption Sometimes a plaintext, with a proof of proper decryption, will be supplied to some participants but not published on the BB. We call this *private decryption.*

Mixnet To maintain the confidentiality of each voter's ballot, a mixnet is used to mix it with others. It takes a set of encrypted ballots and outputs a shuffled set of re-encrypted ballots, making it challenging to link any specific encrypted ballot in its output to any of its input encrypted ballots. The mixnet consists of multiple mixing servers arranged in sequence, each operated by an independent mixing authority, ensuring that as long as at least one server is honest, the confidentiality of the ballots is preserved. The mixnet operation on the input encrypted ballot set $\{E(\mathbf{b_i})\}_i$ is denoted by $Mix(\{E(\mathbf{b_i})\}_i)$.

The mixnet proves that its input $\mathcal{I}$ (a set of vote vectors) is correctly shuffled and re-encrypted to $\mathcal{O}$, without revealing any information about the permutation from $\mathcal{I}$ to $\mathcal{O}$. This proof is denoted by $ZKP_{\mathrm{MIX}}(\mathcal{I}, \mathcal{O})$

Digital signature The digital signature on message m by voter i is represented as $sig_i(m)$. Most often we produce both the message and its signature, which we denote by $auth_i(m)$, meaning $sig_i(m)$ concatenated with m.

Hash function $H(\cdot)$ denotes the hash of message m using a collision-resistant cryptographic hash function (e.g., SHA-256).

Serial Number Each ballot includes a unique serial number sn which is randomly generated. The role of the serial number is to allow one-to-one matching between the paper ballot and its electronic counterpart.

2.2 Risk Limiting Audit

A *Risk Limiting Audit* (RLA) tests whether a trustworthy set of paper ballots implies that the announced election winner(s) won, and corrects the result by a full hand count if not. It is parameterized by a *risk limit* α and guarantees that if the announced election result is wrong, the RLA will progress to a full hand count with probability at least $1 - \alpha$. RLAs were developed by Stark and others and apply to a wide variety of election types and audit situations, including plurality contests, supermajority elections, party-list proportional elections and Instant Runoff Voting. A general framework is given in [12].

The *apparent outcome* is (a set of) announced winner(s). The *actual outcome* is the outcome that would be found if all the ballot papers were correctly counted. A *ballot manifest* is a catalogue describing which paper ballots are present at which physical location. The apparent outcome is supported by a set of *cast vote records* (CVRs), which may or may not accurately reflect the voter's intentions. In this work we mostly concentrate on *ballot-level comparison audits* in which each individually sampled ballot paper is compared with its corresponding CVR. (Using MERGE with other audit styles is discussed in section E.) If the CVR overstates a winner's tally compared with the paper record, it is an *overstatement* (for example, if the CVR is a vote for the winner and the paper ballot is a vote for the loser, it is a two-vote overstatement). If the CVR understates a winner's tally compared with the paper ballot, it is an *understatement*.

An RLA technique that is particularly useful in our setting is the phantoms-to-zombies approach devised by Bañuelos and Stark [2]. This is a way of dealing with ballots that appear in the manifest but cannot be found on paper, a problem that may occur frequently when MERGE ballots are delayed or dropped in the mail. The idea is to "Pretend that the audit actually finds a ballot, an evil zombie ballot that shows whatever would increase the risk value the most." Bañuelos and Stark prove this to be conservative, in the sense that the RLA property still holds, assuming that it would have held had the correct ballot paper been located. The correct evil zombie ballot might be slightly different depending on the kind of RLA, but is generally a vote for the highest loser, or possibly an imaginary vote for all the losers, or (in the case of separate comparisons between the winner and each loser) a vote for whichever loser is being compared.

There are various RLA approaches, each with unique properties, defined by different mathematical functions for deciding when to accept the outcome. The following definition outlines their general functionalities and risk limit properties.

Definition 1. ***RL Functionality***

Setup parameters: *An apparent outcome, a ballot manifest, a risk limit α, and optionally the cast vote records.*

Input: *A sample of ballots (their manifest ID and vote contents), optionally indexed by their cast vote record.*

Procedure: *Perform one of the following actions:*

- *Output "accept" and halt, or*
- *Output "reject" and halt, or*
- *Take a new sample of ballots, combine it with the previous sample, update risk calculations and repeat.*

Risk Limit Property: *For any setup parameters with an incorrect apparent outcome, the probability that the above procedure outputs "accept" over uniformly generated[3] random samples is at most α.*

It is important to understand that applying an RL Functionality does not necessarily imply running a valid Risk Limiting Audit. For example, if the ballot papers are not an accurate representation of the voters' intent, if they have not been securely stored, if the CVRs were not properly committed before the random sample was taken, or the sampled ballots were not honestly randomly generated, the audit may not actually limit risk. The exact definition of a valid risk limiting audit is outside the scope of this paper. We can, however, prove that our RL functionality maintains the risk limit property, which implies that it can be fitted in to an existing Risk Limiting Audit process.

Remark 1. The risk limit property holds regardless of how the wrong outcome is constructed: the adversary may choose any margin, or any way to distribute the wrong CVRs, but must commit to CVRs before samples are chosen randomly.

[3] Some techniques exist for specific kinds of non-uniform sampling, and could easily be incorporated into MERGE, but are not considered in this paper.

3 Threat model and trust assumptions

Here, we define our trust assumptions as follows:

1. The local electoral authorities maintain a list of CAC IDs assigned to registered eligible voters within their county.
   - Each voter's CAC card securely stores the corresponding private key.
   - The CAC Certificate Authority issues trustworthy certificates linking each CAC ID to its public key(s).
   - The local electoral authorities validate the CAC certificate for each signed ballot posted to the BB.[4]
2. Each voter verifies that the paper ballot accurately reflects their intention.
3. Each printed signature on the sticker is verified before sending, either by the voter or by some other trustworthy assistant at the Remote Voting Center.
4. The mailing addresses on the envelopes are correct[5].
5. The voter instructions have to come from some trustworthy source, e.g. a poster on the wall, and not from the BMD itself.

The below additional assumptions are relevant only for privacy.

6. The voter remains hidden from others within the confines of the voting booth. The privacy adversary's visibility is restricted to observing
   - the BB,
   - other pieces of evidence (remembered or captured by the voter), which might be susceptible to forgery.
7. Fewer than a pre-determined threshold of tallying authorities are dishonest.
8. At least one of the mixing authorities is honest.

Integrity also depends on an accurate count of the number of voters who participated—the procedures for achieving this are described in section B.3.

3.1 Attacker model

The adversary may control all computers (except at least one chosen for verification) and do any polynomial-time computation. Any envelope, mailed by a legitimate sender, may be selected by the adversary to experience extended delays or even go missing. However, if delivered, the envelope and its contents are unaltered. The adversary is unable to read the contents of a mailed envelope.

Consequently, the attacker can manipulate a compromised voting machine with the voter's CAC card to sign any arbitrary message. We therefore do not

[4] In principle, the certificates could be posted on the BB with the signatures. However, in practice CAC certificates contain significant personal information that precludes public distribution. We therefore need to assume that they are verified by local authorities but not the public.

[5] This assumption can hold if: the sender knows the intended destinations, there is a supply of preprinted trustworthy envelopes, or an authority verifies the address printed on the envelope by the BMD.

make it harder for this *internal* attacker to stuff a valid-looking paper ballot into the paper mail than it would have been with traditional postal mail. If the jurisdiction was diligently checking each voter's handwritten signature, or if some other form of registered or secure mail is being used, our protocol does not undermine it, but nor do we add to the defences. So we simply assume that *something* prevents the internal attacker from adding fraudulent mail ballots.

MERGE does not claim to defend against paper ballot stuffing.

4 MERGE protocol

This section describes the protocol. The example given here uses a BMD to print a paper record, but the protocol would work just as well with a hand-marked paper ballot interpreted by a scanner. In that case, the "voting computer" would be a scanner with the capacity to produce an encrypted record and upload it to the BB. The serial numbers could either be generated by the scanner, or printed in advance. We will use the term "voting computer" or "voting machine" to mean either a BMD or a scanner with computation and communication capacity.

4.1 Voting process for voter i

The voting process from the voter's perspective is described below.

1. The voting machine displays the options to voter i.
2. The voter interacts with the machine to make their selection, vote $\mathbf{b}_i$.
3. The voting machine assigns a unique serial number sn_i to this vote. The voting machine produces:
   - a signed, encrypted electronic record and serial number, with a proof of ballot validity, for the BB:
   $$BB \hookleftarrow auth_i(E(sn_i), E(\mathbf{b_i}), ZKP_{\text{VALID}}(\mathbf{b_i})),$$
   - a plain paper record of the ballot $(sn_i, \mathbf{b_i})$, which is placed in an envelope,
   - a sticker that includes an address, a traditional mail tracking number, a space for a pen-and-ink signature (if required by regulation—not relevant to our cryptographic protocol) and a digital signature $auth_{Voting\ Computer}(Id_i, Id_e, H_i)$, where Id_i is voter i's unique identifier, Id_e is the election identifier, and H_i is the hash value created from the digital record that is stored for voter i, i.e. $H_i = H(auth_i(E(sn_i), E(\mathbf{b_i}), ZKP_{\text{VALID}}(\mathbf{b_i})))$.

   The voter must verify that her plaintext printout matches her intention, then stick the sticker on her envelope and mail it. Someone also needs to use an electronic device, whether their own or a third party's, to
   - verify the digital signature and the signed data on the sticker, and
   - the "BB Inclusion Check" or "Sticker BB Upload" (section B.3).

After the voting period ends, electronic records undergo a series of processing steps. Additionally, paper ballots are sent to the Local Counting Center for auditing purposes. A diagram illustrating the whole process is in Figure 2. Details of all processing steps will be provided in the following sections.

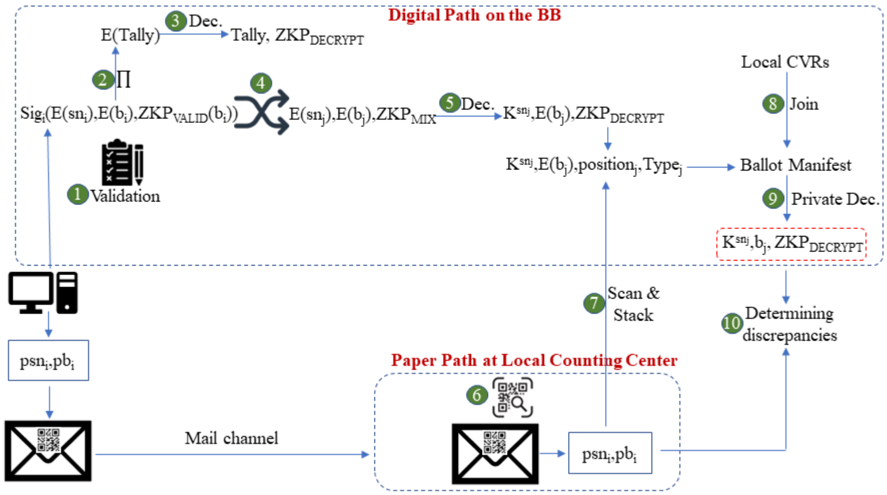

Fig. 2: The whole process for MERGE based on ballot comparison. Each Local Counting Center's votes are dealt with separately—the diagram shows the process for only one Local Counting Center.

4.2 Digital path

We explain digital processing of electronic ballots for one Local Counting Center's votes. Every other Local Counting Center's data is processed similarly—they do not interact because their results need to be processed by the appropriate Local Counting Center, even for statewide contests.

At the conclusion of the voting process, each jurisdiction processes the votes on the BB corresponding to CAC IDs registered in that jurisdiction.

The digital record undergoes the following processing steps.

4. The validity of the signature and ZKP for each electronic ballot on the BB is verified. It is also verified that no two ballots on the bulletin board contain a common ciphertext. If this occurs, it indicates serious misbehaviour by a voting computer—the process stops. If any two valid ballots come from the same CAC ID, the first one is selected and the rest are withheld from any further processing.[6]
5. The digital record on the BB now contains the information of all n voters. The record with index i denotes the ballot from voter i, where $i = 1 \dots n$.

$$BB \hookleftarrow \mathcal{D}_1 := auth_i(E(sn_i), E(\mathbf{b_i}), ZKP_{\text{VALID}}(\mathbf{b_i})) : i = 1 \dots n$$

[6] It does not matter which is selected, as long as only one is chosen, and all n selected for processing by a given jurisdiction are unambiguously marked on the BB.

6. The encryption of the total tally is calculated and published on the BB

$$BB \hookleftarrow E(\mathbf{Tally}) := \prod_{i=1}^{n} E(\mathbf{b_i}).$$

7. $E(\mathbf{Tally})$ is decrypted and published on the BB along with the proof of the correct decryption.

$$BB \hookleftarrow E(\mathbf{Tally}), \mathbf{Tally}, ZKP_{\text{DEC}}(\mathbf{Tally}, E(\mathbf{Tally}))$$

   (In some jurisdictions, this value may be sensitive because of a small set of MERGE voters. See Appendix E for variations that allow public verifiability without publication of the separate tally.)
8. The votes and serial numbers are extracted from $\mathcal{D}_1$ and then mixed—denote this by $\mathcal{D}_2 := E(sn_i), E(\mathbf{b_i}) : i = 1 \ldots n$. The mixed pairs along with the proof of the correct mix operation are published on the BB. So, at the end of this step, we have the following data on the BB:

$$BB \hookleftarrow \mathcal{D}_3 := (E(sn_j), E(\mathbf{b_j})) : j = \pi(i), j = 1 \ldots n.$$
$$BB \hookleftarrow ZKP_{\text{MIX}}(\mathcal{D}_2, \mathcal{D}_3)$$

   where π is the (secret) permutation applied by the Mixnet.
9. Serial numbers are decrypted and the proof of the correct decryption is published. At the end of this step, we have the following data on the BB.

$$BB \hookleftarrow \quad \mathcal{D}_4 := (K^{sn_j}, E(\mathbf{b_j}), ZKP_{\text{DEC}}(K^{sn_j}, E(sn_j))) : j = 1 \ldots n.$$

   It is also checked that there are no duplicate serial numbers. Any occurrence of duplicate serial numbers indicates significant problems with a voting computer—if there are any, the process should stop.

4.3 Paper path

At the end of the voting process, the ballot papers are sent to the Local Counting Center individually in separate envelopes. Each paper record undergoes the following processing steps.

10. If the jurisdiction has an existing process of scanning traditional handwritten signatures, this would apply at this point. Envelopes with invalid signatures must be set aside and handled properly.
    Each incoming envelope's sticker is scanned and its corresponding digital signature is verified. It is necessary to check that the data being signed matches the corresponding data on the BB for this voter, that the voting computer's signature is valid, and that the CAC ID is registered to vote in this jurisdiction. Then, the envelope is accepted and its corresponding record on the BB is marked as received. If any of these verifications fails, the process

continues but the envelope is set aside unopened in a stack called "Rejected Envelopes" for further investigations. If the signature is valid but another validly signed envelope has already been received from the same voter, it is set aside in a stack called the "Eligibility Problem" stack.

For all validly signed envelopes (excluding the ones inside the "Eligibility Problem" stack), the envelope contents are removed and physically shuffled, similar to standard postal voting procedures.

11. Each incoming ballot's serial number psn is scanned. For each serial number, K^{psn} is computed and it is checked if there is a digital ballot with a matching K^{sn} on the BB.[7] Then, the location of the paper ballot is appended to the corresponding digital ballot on the BB.[8] Each digital ballot is accompanied by a parameter called TYPE where TYPE = "not matching", TYPE = "one-to-one" and TYPE = "duplicated" respectively indicate if there exist zero, one or multiple paper ballots with the matching serial number. We then have the following data on the BB:

$$BB \hookleftarrow \mathcal{D}_5 := \{(K^{sn_j}, E(\mathbf{b_j}), \{\textsc{pos}_{j'}\}_{j'}, \textsc{Type}_j) : \quad j = \pi(i), j' = \pi'(i), j = 1 \ldots n.\}$$

The permutation π' represents a physical shuffle of paper ballots, and $\textsc{pos}_{j'}$ denotes the location of the associated paper ballots in storage. For digital ballots with TYPE = "duplicated" the positions of all matching ballots are recorded on the BB. For digital ballots with TYPE = "not matching", the parameter POS is null.

Any paper ballot with no matching K^{sn} on the BB is set aside. Call this the "paper-only" stack. These need to be dealt with outside the cryptographic protocol.

4.4 RLA

At the Local Counting Center the usual local Cast Vote Records (CVRs) are joined with the MERGE CVRs for auditing purposes, and the ballot manifest is updated to include both kinds of records. When a local ballot is sampled, it is retrieved and dealt with as usual. When a MERGE ballot is sampled, its discrepancy is determined according to the guidelines outlined below. Then RLA statistics are updated accordingly, in exactly the same way for remote ballots as they would be for local ones. Local officials make local decisions to accept the result or escalate to a larger sample, according to whatever RLA calculations they usually conduct.

[7] Because of the way ElectionGuard encryption works, the number that naturally drops out after decryption is K^{sn}, not sn. Because we care only about exact matches, it does not matter whether we work in the exponential or plain form. We do not assume that K^{sn} hides sn, and do not need to hide the values of sn, because these are not publicly associated with the voter.

[8] This is its physical location in paper ballot storage, as in a ballot manifest.

Determining discrepancies Assigning discrepancies to digital ballots is primarily determined by the stack to which the ballot belongs.

12. If the type for the selected ballot is either "one-to-one" or "duplicated", its encrypted vote $E(\mathbf{b})$ is privately decrypted to $\mathbf{b}$ and, alongside the proof of correct decryption, is supplied to all auditors and observers in the Local Counting Center, who verify the NIZKPs. They have access to the following data for the selected ballot: **Observers:** $K^{sn_j}, E(\mathbf{b_j}), \mathbf{b_j}, ZKP_{\text{DEC}}(\mathbf{b_j}, E(\mathbf{b_j})), \{\text{POS}_{j'}\}_{j'}$ Then, its discrepancy is determined as follows.
    - Identify the paper ballot situated at position $\text{POS}_{j'}$ and read its serial number *psn*. If K^{psn} does not match K^{sn_j}, output "serial number error" and terminate.
    - Compare the vote on the paper $\mathbf{pb}$ to $\mathbf{b_j}$ and record the discrepancy, exactly like any other RLA.

    For ballots of type "duplicated", the above 2-step procedure is performed for all j' values and the final discrepancy is set to the *minimum* discrepancy among different j' values for the digital ballot. That is, in case of several paper ballots, one of the ballots with lowest discrepancy value is selected.
    Any occurrences of the "serial number error" indicates a significant problem with the scanning device and hence necessitates restarting the entire process from step 7 onward using another scanning device.
13. For ballots with TYPE = "not matching", the encrypted ballot is privately decrypted and, alongside the proof of correct decryption, is supplied to all auditors and observers. Then, the discrepancy is determined by setting it to the maximum possible value for such a ballot, or in other words, employing a worst-case paper assumption for the given ballot.

A summary of all verification steps is provided in Appendix B.3, with details of error handling in Appendix B.4.

5 Security analysis

Verifiability We begin by outlining the audit process in an ideal scenario featuring trustworthy paper ballots and a one-to-one correspondence between each paper ballot and its electronic record. A subset of voters in the ideal world is controlled by the adversary, but other voters follow the voting instructions.

Next, we progressively transform this ideal world into our protocol (i.e., the Real World) through a series of intermediate worlds. At each stage, we establish the validity of the following statement: *any combination of digital and paper ballots in the newly defined world corresponds to a set of digital and paper ballots in the present world. This correspondence ensures identical (1) reported totals, (2) number of voters, and (3) honest voters. Furthermore,*

- *The digital ballots in the newly defined world are a permutation of those in the present world, or*

– *For any ballot selected for the RLA in the present world, the discrepancy value is equal to or smaller than the discrepancy value of its corresponding ballot in the newly defined world.*

These statements, based on Remarks 1 and 2 respectively, imply that for any given risk limit, when the announced outcome is wrong, if the RLA functionality with any given sample set accepts the announced outcome in a newly defined world, it will accept the corresponding outcome in the present world. The intuition is that inconsistencies between CVRs and paper records are handled in such a way that the overall risk value does not decrease from one scenario to the next. Since the last defined world is the same as the real MERGE protocol, the proof shows that it has no greater likelihood of accepting a wrong result than an 'ordinary' RLA conducted in the ideal world on the trustworthy paper ballots with a one-to-one correspondence between each paper ballot and its CVR. A precise statement and proof are contained in Appendix B.3.

Privacy MERGE does not provide ballot privacy, because the adversary in this model might drop envelopes and thus use differencing attacks to infer individual votes. For example, if the adversary drops one ballot, labelled on the sticker as Alice's, then the adversary knows whose vote is the unique one with no match. Since there is a non-zero probability that this is audited and opened, Alice's privacy cannot be guaranteed. More generally, since the RLA process may lead to a full manual recount of the arrived paper ballots, differencing attacks are always possible. Appendix C.2 examines the protocol's privacy, and proves that when the attacker cannot drop ballots the protocol satisfies the definition of privacy given in [6]. Appendix C.3 contains a discussion of receipt freeness.

6 Conclusion and future work

Usable voter-verification—without undermining coercion resistance—remains the major open problem in this field. Using paper requires transmitting it back to the counting location; using cryptography demands that the voter successfully performs one of the human-computer-interaction protocols which, despite ingenious recent advances, remain very hard for voters to run successfully. Our protocol combines plain paper verification with cryptographic assurance to get the best properties from each.

Bibliography

[1] Andrew W Appel, Richard A DeMillo, and Philip B Stark. Ballot-marking devices cannot ensure the will of the voters. *Election Law Journal: Rules, Politics, and Policy*, 19(3):432–450, 2020.

[2] Jorge H Bañuelos and Philip B Stark. Limiting risk by turning manifest phantoms into evil zombies, 2012. https://arxiv.org/abs/1207.3413.

[3] Josh Benaloh and Michael Naehrig. Electionguard specification v2.0. 2023.

[4] Josh Benaloh, Michael Naehrig, Olivier Pereira, and Dan S. Wallach. ElectionGuard: a cryptographic toolkit to enable verifiable elections, 2024. Cryptology ePrint Archive, Paper 2024/955.

[5] Josh Benaloh, Philip B Stark, and Vanessa Teague. Vault: Verifiable audits using limited transparency. *Proceedings of E-Vote ID*, 2019. https://www.stat.berkeley.edu/~stark/Preprints/vault19.pdf.

[6] David Bernhard, Véronique Cortier, David Galindo, Olivier Pereira, and Bogdan Warinschi. Sok: A comprehensive analysis of game-based ballot privacy definitions. In *2015 IEEE S&P*, pages 499–516. IEEE, 2015.

[7] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *Advances in Cryptology–ASIACRYPT 2012*, pages 626–643. Springer, 2012.

[8] Henri Devillez, Olivier Pereira, and Thomas Peters. Verifiable and private vote-by-mail. *Cryptology ePrint Archive*, 2024.

[9] Benjamin Fuller, Abigail Harrison, and Alexander Russell. Adaptive risk-limiting comparison audits. In *2023 IEEE S & P*, pages 3314–3331. IEEE, 2023.

[10] Philip Kortum, Michael D Byrne, and Julie Whitmore. Voter verification of ballot marking device ballots is a two-part question: Can they? mostly, they can. do they? mostly, they don't. *Election Law Journal: Rules, Politics, and Policy*, 20(3):243–253, 2021.

[11] Ronald L Rivest. On the notion of 'software independence' in voting systems. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881):3759–3767, 2008. https://people.csail.mit.edu/rivest/pubs/RW06.pdf.

[12] Philip B Stark. Sets of half-average nulls generate risk-limiting audits: Shangrla. In *International Conference on Financial Cryptography and Data Security*, pages 319–336. Springer, 2020. ArXiv: https://arxiv.org/abs/1911.10035.

[13] Philip B Stark. Overstatement-net-equivalent risk-limiting audit: Oneaudit. In *Financial Cryptography and Data Security - Voting Workshop*. Springer LNCS 13953, 2023. https://arxiv.org/abs/2303.03335.

[14] Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In *Progress in Cryptology–AFRICACRYPT 2010: Third International Conference on Cryptology in Africa, Stellenbosch, South Africa, May 3-6, 2010. Proceedings 3*, pages 100–113. Springer, 2010.

A Optimizations for multiple contests

Up to now, we have assumed that the election consists of a single contest, resulting in the vote vector $\mathbf{b} = b_{n-1}\|\cdots\|b_0$ containing one bit per candidate within that contest. Therefore, the encrypted vote vector $\mathbf{e}$ is computed as:

$$\mathbf{e} = E(\mathbf{b}) = (E(b_{n-1}), \cdots, E(b_0))$$

To expand our design to accommodate multiple contests while maintaining a common ballot style, we can consider $\mathbf{b}$ as a concatenation of multiple vote vectors, with each vote vector corresponding to a distinct contest. For example, in an election featuring two contests, we express $\mathbf{b}$ as $\mathbf{b_1}||\mathbf{b_0}$, where $\mathbf{b_0}$ and $\mathbf{b_1}$ represent the vote vectors for contests 0 and 1, respectively. The encrypted vote vector $\mathbf{e}$ is accordingly computed as

$$\mathbf{e} = E(\mathbf{b}) = (E(\mathbf{b_1}), E(\mathbf{b_0}))$$

In this configuration, the overall process remains largely unchanged compared to a single-contest election, with the exception that NIZK proofs and discrepancy assignments must be conducted separately for each contest on a ballot.

Furthermore, to extend our design to accommodate elections with multiple ballot styles, it's required to store the ballot style in plaintext alongside each ballot. This is necessary for tallying the ballots within each contest. We must also ensure that the ballot style for each ballot undergoes the same mixnet process as the ballot itself and is presented in plaintext format before the RLA. Otherwise, if the digital ballot that is selected for the RLA lacks a corresponding paper ballot, it would be unclear which contest's RLA statistics should be updated.

Consider $\mathbf{b} = \mathbf{b_{k-1}}||\cdots||\mathbf{b_0}$, representing a set of k distinct contests, where each $\mathbf{b_i}$ is an n_i vote vector representing the i^{th} contest. The total number of options in total is denoted as n, where $n = \sum_i n_i$. In the typical scenario, we would require n encryption values to represent $\mathbf{e} = E(\mathbf{b})$. Then, we conduct the tally and the RLA process separately for each contest. While this approach offers flexibility, it also entails a higher number of encryption values per ballot, namely n, which can potentially complicate and slow down the mixnet process. Thus, we provide some optimizations below.

Let $\mathbf{e}$ comprise n ElGamal encryption values corresponding to an n-bit vote vector. To accelerate the mixnet process, following the tally phase, we apply the following COMP transformation to $\mathbf{e} = (e_{n-1}, \cdots, e_0)$, with each $e_i = (\alpha_i, \beta_i)$, in order to compact it into a single ElGamal encryption value and speed up the mixing process:

$$e' = \text{COMP}(\mathbf{e}) = (\prod_i \alpha_i^{2^i} \bmod p, \prod_i \beta_i^{2^i} \bmod p)$$

Our assumption here is that n is smaller than the size of the group $|q|$ in bits. For larger vote vectors, we must partition the vector $\mathbf{e}$ and construct separate ciphertexts for each segment. While this approach is highly efficient, it lacks flexibility for RLA purposes, as outlined below.

Due to the way ElectionGuard encryption works, the decryption output is $K^{\mathbf{b}}$. For large n (e.g., $n = 100$), finding $\mathbf{b}$ via exhaustive search is infeasible. Normally, if such a ballot is selected for an RLA, $\mathbf{b}$ is derived from the paper, and $K^{\mathbf{b}}$ is compared with $\mathbf{e}$. If these values match, the discrepancy for all contests on that ballot is 0. However, if they differ, or if a digital ballot with TYPE = "not matching" is selected for the RLA, then discrepancy values for all contests on that ballot must be considered as the maximum possible value (e.g., +2 for a plurality voting).

B Protocol specification

B.1 ElectionGuard details

To achieve a decentralized architecture, ElectionGuard relies on the collaboration of multiple entities known as 'guardians,' jointly responsible for protecting voters' privacy. Each guardian independently generates its own public-private key pair. These individual public keys are then combined to derive the election public key, crucial for encrypting all voters' selections. Subsequently, in the decryption phase, each guardian computes a verifiable partial decryption, resulting in a full verifiable decryption process.

To address scenarios where certain guardians may be unavailable, a robust cryptographic mechanism is employed. Guardians cryptographically share their secret keys, ensuring that a pre-defined threshold quorum guardians is required to achieve full decryption. See [4] for details of the key generation protocol.

MERGE follows the ElectionGuard 2.0 specification for primitives, as described in [4]. We give a brief overview of the encryption scheme here.

B.2 Notations and building blocks

The selections of voter i, choosing from m candidates, are represented as a binary vector $\mathbf{b_i}$ of length m, consisting of zeros for unselected options and ones for selections. In cases where it is clear and doesn't cause confusion, the index i can be omitted for simplicity. For example, Alice is another way to represent the vote vector $\mathbf{b} = [1\ 0\ 0]$ in a contest with the following candidate list [Alice Bob Charlie].

The encryption of the vector $\mathbf{b}$ is denoted as $\mathbf{e} = E(\mathbf{b})$, wherein each entry $\mathbf{e}_j$ for $j = \{1, \cdots, m\}$ represents the encryption of the corresponding vote entry $\mathbf{b}_j$.

Encryption scheme In ElectionGuard, encryption of votes is performed using a variant exponential form of the ElGamal cryptosystem. The cryptosystem parameters (p, q, g) are defined as follows: p is a prime number equal to $2kq + 1$, where q is also a prime number, and g is a generator of the order q subgroup of $\mathbb{Z}_p^*$. (The ElectionGuard specification requires particular values of p and q.) To generate a public-private key pair, a private key $s \in \mathbb{Z}_q$ is randomly selected.

The corresponding public key, K, is then computed as $K = g^s \mod p$ and made public. To encrypt a vote b, a nonce ξ is selected randomly such that $0 \leq \xi < q$. Then, encryption of b, denoted by $E(b)$, is computed as

$$E(b) = (g^\xi \bmod p, K^b \cdot K^\xi \bmod p) = (g^\xi \bmod p, K^{b+\xi} \bmod p).$$

The entity possessing the corresponding secret key s can decrypt (α, β) as $\beta/\alpha^s \mod p = K^b \mod p$. If b is sufficiently small, its value can be derived from K^b using an exhaustive search. This encryption scheme is additively homomorphic. Namely,

$$E(b_1, \xi_1) \cdot E(b_2, \xi_2) = E(b_1 + b_2, \xi_1 + \xi_2)$$

We can use the following equation to re-encrypt the ballot b

$$E(b, \xi_1) \cdot E(0, \xi_2) = E(b, \xi_1 + \xi_2)$$

Hash function The function H that is used in this whole section is the same as one specified in ElectionGuard, which is based on HMAC-SHA-256. The function H takes two inputs, B_0 and B_1, where B_0 is 256 bit long and corresponds to the key in HMAC and B_1, which is of arbitrary length, is the actual input to the HMAC. These inputs are separated by a semicolon. If B_1 is comprised of multiple elements (e.g. $B_1 = a||b||c$), then these elements are separated by commas(e.g. $H(B_0; a, b, c)$).

Zero knowledge proofs We use the following ElectionGuard zero knowledge proofs, which are assumed to be paramaterized by the ElectionGuard group parameters (p, q, g).

- Distributed proof of decryption correctness, used by the guardians to decrypt output ciphertexts.
- $\mathsf{KnowDlog}(K)$: a proof of knowledge of the discrete log of K base g (that is, of x s.t. $K = g^x \bmod p$).
- $\mathsf{EqDlogs}(x, y, X, Y)$: a proof of equality of discrete logs.
- **Proof that (a, b) is an encryption of an integer in the range $0, \cdots, L$**
- **Proof that $(a, b) = (g^\xi, K^\xi)$ is an encryption of zero or one**
- **Proof that $(\alpha, \beta) = (g^\xi, KK^\xi)$ is an encryption of zero or one**

See [4] for details of these zero knowledge proofs. These are applied for our “ZKP of proper ballot construction”.

B.3 Verification Summary

There are three separate verification stages, which can be verified by different people: cast-as-intended verification, which is mostly done by the voter, universal (BB) verification, which can be done by any member of the public, verification by observers at the Local Counting Center that the data on the BB matches the ballot papers and the data included in the tally and the RLA matches the BB. Each of these is detailed below. First, however, there must be a trustworthy count of the total number of participants.

Counting the number of votes & creating a trustworthy ballot manifest
As voters cast their ballots using their CAC cards, the digital signature associated with each vote on the BB offers a dependable method to authenticate the voter. The purpose of the digital signatures is to complicate ballot stuffing—only ballots (either electronic or paper) with a valid accompanying digital signature will be accepted. The voting computer's digital signature needs to be explicit on the mail sticker, but is implicit on the BB because that channel is already authenticated.

However, there is still the possibility of an attack in which a malicious voting computer colludes with some other machine that (at some point) has CAC access, in order to fabricate an apparently-valid digital vote from a voter who did not intend to vote (and did not appear at the polling place). Since this is electronically indistinguishable from valid voting, it must be defended by human procedures.

– Officials at the Remote Voting Center must keep a count of the total number of properly completed votes.

It is a requirement of any RLA that there must be a record of how many ballot papers there are, derived independently of the scanners that are being audited. Otherwise, if the paper records are also delayed or dropped, there is no way to know how many ballots have been dropped, and consequently what the implications for the accuracy of the election result might be.

There also needs to be a way to verify that ballots have not been dropped from the electronic record. This needs to be supported by one of the following practical procedures.

– Officials at the Remote Voting Center could keep a record of who voted there and verify that it (eventually) matches the set of valid signatures on votes on the BB.
– "BB Inclusion Check": Every voter (or their delegate) would need to verify the inclusion of their vote on the BB. The envelope sticker can be used as participation evidence.
– "Sticker BB Upload": Every voter (or their delegate) would upload the sticker's data to the server and receive a signed receipt. When possible, the server would send it to the BB. Then, everyone can check the presence of a corresponding vote on the BB for every uploaded sticker.

Options 2 and 3 are variations of the same concept: the actual BB upload might take a long time, so we ensure that the voter gets an “inclusion promise”, which can be used as evidence that she voted. In the case of “BB Inclusion Check”, the envelope sticker itself is the “inclusion promise”; in the case of “Sticker BB Upload”, the server’s signature on that value is an inclusion promise from the server. This allows for evidence of different kinds of malfeasance from different parties, and requires different forms of verification. Using either of the stated methods to process the sticker’s data does not compromise privacy and may be done by voters, officials or bystanders.

Cast-as-intended verification (by the voter)

1. Verify that the ballot paper matches the intended vote.

Cast-as-intended verification (by the voter or anyone else at the Remote Voting Center)

2. Verify the digital signature on the sticker stuck onto the envelope.
3. For “BB Inclusion Check”: Verify that a properly-signed vote corresponding with the sticker’s data is (eventually) on the BB,
4. For “Sticker BB Upload”: Verify that the sticker’s data appears on the BB.

BB transcript verification (public)

1. For each vote on the BB:
   (a) Verify voter’s digital signature
   (b) Verify ZKP of proper ballot construction
2. Verify that no two ballots on the bulletin board contain a common ciphertext (only necessary for privacy. See Section C.2).
3. Verify mixing ZKP
4. Verify decryption of all serial numbers sn_i.
5. Verify that each sn_i is unique.
6. Verify aggregation of $E(\mathbf{Tally})$.
7. Verify the decryption **Tally** of $E(\mathbf{Tally})$.
8. For “Sticker BB Upload”: verify all uploaded sticker digital signatures and verify the presence of a corresponding ballot on the BB for each sticker.

Verification at the Local Counting Center

1. Verify digital signatures on the stickers, and verify that they accurately sign data that corresponds to the information on the BB (i.e., with correct Id_i, Id_e and H_i). (By both election authorities and auditors at the Local Counting Center. Auditors can use their own electronic devices)
2. Verify that the *Eligibility Problem Stack* includes all (and only) duplicate envelopes from the same voter.

3. Verify that the *Rejected Envelopes Stack* includes all (and only) envelopes with the invalid signatures.
4. Verify that **Tally** is properly added to the local CVRs or tallies.
5. Verify $ZKP_{\text{DEC}}(,)$ for each ballot that is privately decrypted during the RLA process.
6. Verify that discrepancies are properly determined and then correctly incorporated into the RLA.
7. Verify there are no “serial number error”s.

B.4 Failure handling

In this section, we clarify the procedure that must be adopted if each aforementioned verification fails.

Cast-as-intended verification Item 1: If the paper ballot does not match the voter's selections, this could be attributed to a voting machine failure or a mistake by the voter. In such cases, the voter has the option to either retry the voting process or abandon it.

In either case, the following steps are taken:

- An authority records the voter's identity and the associated sticker, which is considered invalid from that point onward. If an envelope with an invalidated sticker arrives at the Local Counting Center, it is rejected. This recorded information serves as a reference in case the voter later claims possession of a sticker whose corresponding ballot is not found on the BB.
- The invalidated digital ballot is marked on the BB.

Furthermore, if the voter chooses to retry voting, the voting machine captures the new digital ballot as their vote and generates a corresponding paper ballot and sticker.

Item 2: If the digital signature on the sticker is not verified, it is the voting machine's fault.

Item 3: If a properly-signed vote, corresponding to the sticker's signature, from this voter does not (eventually) appear on the BB, it is the voting machine's fault.

BB transcript verification (public) Items 1a and 1b: If for any digital ballot on the BB, digital signature or $ZKP_{\text{VALID}}()$ is not successfully verified, it is the voting machine's fault.

Item 2: Given the negligible probability that any two ciphertexts are identical, if a ciphertext matches an already published one, it is overwhelmingly likely to be the voting machine's fault (though if the duplicated ciphertexts come from two different voting computers, it is not clear which one is at fault, unless there is reliable timing information).

Item 3: If mixing ZKP is not successfully verified, it is the back-end server's fault. The RLA process must be performed after resolving the issue.

Item 4: If decryption of each encrypted serial number is not successfully verified, it is the back-end server's fault. The RLA process must be performed after resolving the issue.

Item 5: Given that it is of negligible probability that any two randomly generated serial numbers are equal, if each serial number is not unique, it is highly likely the voting machine's fault.

Item 6: If computation of $E(\mathbf{Tally})$ is not successfully verified, it is the back-end server's fault. Decryption of $E(\mathbf{Tally})$ and then adding the decryption result to tallies from other systems must be performed after resolving the issue.

Item 7: If decryption of encrypted tally is not successfully verified, it is the back-end server's fault. Decryption of $E(\mathbf{Tally})$ and then adding the decryption result to tallies from other systems must be performed after resolving the issue.

Item 8: For the case where "Sticker BB Upload" is used, if there is not a corresponding ballot for each uploaded sticker, it is the voting machine's fault.

Verification at the Local Counting Center Item 1:

- If the sticker on any envelope carries a valid digital signature from the voting machine, but the corresponding signed data does not appear on the BB, it is the voting machine's fault (unless the sticker has already been invalidated as part of the Cast-as-intended verification in item 1).
- If two stickers carry different valid signatures from a voting machine on the same message, that is the voting machine's fault.
- If two stickers carry valid signatures from different voting machines on the same voter' data, that is the voter's fault.
- If any two stickers contain a same valid signature by a voting machine, that could be considered a fault of anyone with access to the sticker (e.g., the voting machine, the voter or even someone working in the postal system).

Items 2 and 3: Any fault in the envelope stacking process is either due to the fault of the involved authorities or the fault of the device used for the signature verification.

Item 4: Any fault is the involved authorities' fault.

Items 5: If decryption of any ballot or aggregation of any sets of ballots and then its decryption are not successfully verified, it is the back-end server's fault.

Item 6: If discrepancies are not correctly determined and then integrated into the RLA computations, it is the involved authorities' and the RLA calculator's fault, respectively.

Item 7: Any "serial number error" is the scanning device's fault.

Item 7: Any fault in counting the number of ballots in the ballot box is the involved authorities' fault.

C Security Proofs

C.1 Verifiability

The proof relies on assuming that the RLA function satisfies the following definition.

Definition 2. *An RLA functionality is* monotonic *if for any given setup parameters with an incorrect apparent outcome and any input sample set, an increase in the discrepancy*[9] *value of any sampled ballot, while other discrepancy values remain fixed, does not change the output from 'reject' to 'accept'.*

Remark 2. All RLA functions in general use are monotonic, and we will assume monotonicity in this paper.

We now develop the main result, described informally in section 5, that our protocol does not give the attacker a non-negligibly higher probability of passing verification for a wrong election result than it would in a plain (ideal) RLA.

Ideal World Consider an Ideal World where every voter visits a local polling place to verify their paper ballot and puts it in a ballot box. A subset of voters is under the control of the adversary. Consequently, the adversary determines the participation status of each in the voting process. The adversary can also influence these voters, leading them to deviate from the prescribed voting process instructions.

The voting machine records a digital ballot for each voter, where the validity of the digital ballot content is publicly verifiable. The resulting CVRs are accurately incorporated into the local CVRs where authorities at the Local Counting Center verify the correctness of this operation (i.e., Section B.3, Verification at the Local Counting Center, item 4).

In this ideal world, the voter's identifier is recorded on both the paper and its digital record. A precise one-to-one correspondence exists between each paper ballot and its electronic record. This correspondence is achieved relying on a trusted authority who verifies whether the digital ballot for the voter appears on the BB before allowing the voter to put their ballot in the box. In other words, the trusted authority ensures the fulfillment of two critical requirements: (1) the identifiers printed on the paper ballots form a subset of the identifiers recorded on the BB (ensuring inclusion), and (2) the quantity of votes in the ballot box is equal to the number of ballots recorded on the BB (ensuring exclusion). We assume the voting authority also has a way to verify each voter's identity.

The adversary has control over all electronic devices, including the voting computer. Therefore, the voter's selection on each paper ballot might be different from its corresponding digital record. After tallying the digital ballots and reporting the voting outcome, we need to verify if discrepancies between digital and paper ballots are not enough to change the voting outcome. So, we execute an RLA process by unsealing the ballot box and randomly sampling from the paper records and using the voter's identifier printed on the chosen paper to locate its corresponding digital record. Then, discrepancies are properly determined and then correctly incorporated into the RLA statistics where authorities at the Local Counting Center, verify the correctness of the operation (i.e., Section B.3, Verification at the Local Counting Center, item 6). To guarantee that

[9] The difference between the voter-verified paper record and the reported electronic results for a specific ballot

an incorrect voting outcome will lead to a manual tally with a predefined risk limit, each voter must check if the selections, printed on their paper ballot, is correct (i.e., Section B.3, Cast-as-intended verification by the voter, item 1).

Ideal World Game The adversary $\mathcal{A}$ wins the ideal-world game if in the Ideal World,

- the apparent election result, according to the (MERGE and regular) CVRs, is different from the actual result, according to the (MERGE and regular) paper ballots, and
- the RLA procedure with a configured risk limit α (including the MERGE cryptographic verification) confirms the result with a probability non-negligibly greater than α, where the probability is taken over the random ballot selections of the RLA, randomization in cryptographic algorithms, and the adversary's random choices.

Real World This world resembles our protocol in the Electronic&Drop or Electronic-only model, with an adversary with the following capabilities:

- Controlling a subset of voters, causing them either not to vote or to deviate from established procedures.
- Printing arbitrary serial numbers on paper ballots.
- Controlling the voting computer, thereby recording digital records that are inconsistent with the voter's selections.
- Dropping some envelopes in the Electronic&Drop model.
- Controlling the scanning device (used in step 11 of the protocol to scan paper ballots' serial numbers), causing it to record arbitrary serial numbers.

The Real World Game is defined similarly to the Ideal World Game, with the distinction that it is set in the Real World.

Our main result is that MERGE does not give the adversary a non-negligibly higher chance of causing the RLA to accept a wrong result, than running the normal RLA without MERGE.

Corollary 1 *Assuming all verification steps (outlined in Appendix B.3) are successfully completed, for any adversary $\mathcal{A}$ who wins the Real World Game, there exists an adversary $\mathcal{A}'$ who wins the Ideal World Game.*

Proof. The proof for the Electronic&Drop model follows directly from Propositions 2 to 9. The proof for the Electronic-only model follows directly from Propositions 3 to 5 and Propositions 6 to 9. These propositions are proven in Appendix C.1. They gradually transform a real-world attack into an equally successful ideal-world attack.

We prove the Propositions required for Corollary 1. This section provides the intermediate worlds and proofs of their equivalence that are used to prove the main verification result, Corollary 1.

Intermediate World 1 Ballots in this world closely resemble those in the Ideal World. However, the RLA is carried out by randomly sampling from the digital records, identifying their corresponding paper ballots, and subsequently adjusting the RLA statistics in accordance with any discrepancies.

Intermediate World 2 In this world, the ballots closely resemble those in Intermediate World 1. However, a unique serial number is recorded alongside the voter's unique identifier on digital ballots. Uniqueness of serial numbers published on the BB can be publicly verified. This serial number effectively replaces the voter's unique identifier on the corresponding paper ballot, serving as an alias for the voter. So, the authority in this world does not check if voters' identifiers printed on the paper ballots form a subset of the those recorded on the BB. The authority still checks if the quantity of votes in the ballot box is equivalent to the number of ballots recorded on the BB. The authority also ensures if the identities, published on the BB, are not spoofed. In this world, there is a potential for a malicious voting machine to deviate from the protocol by printing a permutation of serial numbers (which are published on the BB) on the paper ballots. In other words, although the set of all serial numbers published on the BB are the same as those printed on the paper ballots, the serial number that is assigned to voter i on the BB, might be printed on voter j's paper ballot where $i \neq j$. Consequently, the RLA process in this world unfolds as follows: A digital ballot is chosen at random for the RLA process, and its serial number is utilized to locate the corresponding paper ballot. Subsequently, discrepancies are accurately identified and appropriately integrated into the RLA statistics.

Intermediate World 3 In this world, the ballots closely resemble those in Intermediate World 2. However, dishonest voters in this world might choose not to put their ballot in the ballot box. Also, the authority checks if the quantity of ballots in the box does not exceed the number of ballots on the BB (i.e., Section B.3, Verification at the Local Counting Center, item 7). Authority still ensures if voters identities, recorded on the BB, are not spoofed. Moreover, a malicious voting machine might deviate from the protocol by printing arbitrary, non-matching serial numbers on paper ballots. Consequently, the RLA process in this world unfolds as follows: A digital ballot is randomly selected for RLA and subsequently categorized into one of the following groups, each with its corresponding discrepancy specification, where authorities at the Local Counting Center verify this operation:

– 'not matching': When the serial number of a digital ballot fails to match any serial number in the paper ballot manifest, the discrepancy for the selected ballot is assigned by employing a worst-case paper assumption for the given ballot.
– 'one-to-one': When the serial number of a digital ballot matches a single entry in the paper ballot manifest, its discrepancy is computed accordingly and incorporated into the RLA statistics. It's important to note that there is a one-to-one correspondence between 'one-to-one' digital ballots and paper ballots.

– 'duplicated': When the serial number of a digital ballot matches multiple entries in the paper ballot manifest, the discrepancy between the digital record and each of those multiple entries is computed individually. Among these calculated discrepancies, the minimum discrepancy value is selected for inclusion in the RLA statistics. The paper ballot associated with this minimum discrepancy is identified as the corresponding paper ballot for the digital ballot. In cases where multiple paper ballots share the same minimum discrepancy, one of them is randomly chosen to serve as the corresponding paper ballot for the digital ballot.

Intermediate World 4 In this world, the voting process and the paper path differ from the previous world as follows:

1. There is a trusted party who honest voters can rely on to guarantee the presence of a ballot corresponding to their unique identifier on the BB,
2. Voters place their ballot inside an envelope labeled with their ID and send it to the Local Counting Center,
3. Every paper ballot undergoes the same processing steps as those within MERGE (steps 6 and 7 from the paper path explained in Section 4.3). However, rather than checking the signatures on the stickers, jurisdiction authorities accept any envelope labeled with IDs on the BB.

Intermediate World 5 The only distinction between this world and Intermediate World 4 is that the scanning device might malfunction by recording incorrect serial numbers.

Intermediate World 6 Differences between this world and the Intermediate World 5 are as follows: The serial numbers and the vote contents of digital ballots are initially recorded by the voting machine in an encrypted format. Each digital ballot is also accompanied by a zero-knowledge proof demonstrating the validity of the (encrypted) vote. Subsequently, all digital ballots undergo the same processing steps as those within MERGE (i.e., steps 4 to 9 from the digital path explained in Section 4.2) followed by private decryption in step 9.

Intermediate World 7 In this world, instead of depending on a trusted third party, the voter's digital signature on their digital ballot serves as publicly verifiable evidence that the voter's identity is not spoofed.

Intermediate World x game with $x \in \{1, \cdots, 7\}$ is defined similarly to the Ideal World game, with the distinction that it is set in the Intermediate World x.

Proposition 2 *Given that*

– *successful verification of each digital signature on a sticker by its corresponding voter (i.e., Section B.3, Cast-as-intended verification, item 2),*
– *no reported errors from voters regarding missing votes on the BB (i.e., Section B.3, Cast-as-intended verification, item 3),*

for any adversary $\mathcal{A}$ who wins the Real World Game in the Electronic&Drop model, then there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 7 Game.

Proof. The only distinction between the Real World in the Electronic&Drop model and the Intermediate World 7 is the method used to prevent digital ballot drops. In the Intermediate World 7, a trusted party ensures that voters' ballots are present on the BB. In the Real World, this assurance is achieved through the voting computer's signature on the sticker. So to prove the proposition, we first demonstrate that the voting machine's digital signature $auth_{Voting\ Computer}(Id_i, Id_e, H_i)$, printed on the sticker which is provided to a voter with unique identifier Id_i, proves the voting machine's commitment to publish a digital ballot with the hash value H_i for a voter with the same identifier Id_i in an election with the same election identifier Id_e on the BB.

Consider the opposite in the Real World: a scenario where $auth_{Voting\ Computer}(Id_i, Id_e, H_i)$ does not prove the voting machine's commitment to publish the corresponding digital ballot on the BB. The private key of the voting machine is securely held by the voting machine. If, however, anyone manages to generate a valid digital signature without the voting machine's involvement and hence without access to their private key, it would compromise the security of the digital signature system. Such a situation contradicts our initial assumption regarding the security of the cryptographic primitives employed in the system. Therefore, since honest voters verify the signature on the sticker and they report no errors, the ballots of honest parties are recorded on the BB. Also, due to having a valid sticker, their envelopes, once received, will be accepted at the Local Counting Center. If the serial numbers and vote selections on the paper and digital ballots in the Intermediate World 7 match those in the Real World, then for any sequence of ballots selected for auditing, the number of discrepancies detected by the audit in the Real World will be equal to that in the Intermediate World 7. Consequently, the RLA output in both worlds would be the same.

Proposition 3 *Given that all digital signatures on the BB are successfully verified (i.e., Section B.3, BB transcript verification (public), item 1a), for any adversary $\mathcal{A}$ who wins the Intermediate World 7 Game, then there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 6 Game.*

Proof. The only distinction between the Intermediate World 7 and the Intermediate World 6 is the method used to prevent digital ballot stuffing. In the Intermediate World 6, a trusted party ensures that identities recorded on the BB are not spoofed. But in the Intermediate World 7, the voter's digital signature on their ballot achieves this goal. So, we first demonstrate that the voter's digital signature proves that their identity is not spoofed.

Consider the opposite in the Intermediate World 7: a scenario where a voter's identity on the BB is spoofed. The private key corresponding to each voter's digital signature is securely stored in their smart card, accessible only to the voter. If, however, the voting machine manages to generate a valid digital signature on the BB without the voter actively attempting to cast a vote and hence without access to their smart card, it would compromise the security of the digital signature system. Such a situation contradicts our initial assumption regarding the security of the cryptographic primitives employed in the system. Therefore, if all digital signatures on the BB are successfully verified, voters' identities are not spoofed. Moreover, if the serial numbers and vote selections on the paper and digital ballots in the Intermediate World 6 match those in the Intermediate World 7, then for any sequence of ballots selected for auditing, the number of discrepancies detected by the audit in the Intermediate World 7 will be equal to that in the Intermediate World 6. Consequently, the RLA output in both worlds would be the same.

Proposition 4 *Given that correctness of*

- *all* $ZKP_{\text{VALID}}()$ *proofs (i.e., Section B.3, BB transcript verification (public), item 1b),*
- *all* $ZKP_{\text{DEC}}(,)$ *proofs including:*
  - *decryption of all serial numbers (i.e., Section B.3, BB transcript verification (public), item 4),*
  - *decryption of vote selections for ballots that are selected for the RLA process (i.e., Section B.3, Verification at the Local Counting Center, item 5), and*
  - *decryption of the encrypted tally obtained at step 7 of the digital path (i.e., Section B.3, BB transcript verification (public), item 7)*
- $ZKP_{\text{MIX}}(,)$ *proof (i.e., Section B.3, BB transcript verification (public), item 3), and*
- *aggregation of encrypted ballots at step 6 of the digital path (i.e., Section B.3, BB transcript verification (public), item 6)*

are successfully verified, for any adversary $\mathcal{A}$ *who wins the Intermediate World 6 Game, then there exists an adversary* $\mathcal{A}'$ *who wins the Intermediate World 5 Game.*

Proof. Let $\mathcal{B}$ denote the list of digital ballots recorded on the BB for N voters before initiation of the step 4 in the Intermediate World 6:

$$\mathcal{B} = \{(E(sn_i), E(\mathbf{b_i}) : i \in \{1, \cdots, N\}\}$$

By relying on the security guarantees provided by the $ZKP_{\text{VALID}}()$ protocol, the validation of $ZKP_{\text{VALID}}()$ for all ballots ensures that, for any $i \in \{1, \cdots, N\}$, $\mathbf{b_i}$ represents a valid vote. This crucial property will later enable us to use the same vote selections in the Intermediate World 5. After applying the mixnet, the list transforms to $\mathcal{B}'$:

$$\mathcal{B}' = \{(E(sn'_j), E(\mathbf{b'_j})) : j = \pi(i), i \in \{1, \cdots, N'\}\}$$

Due to validity of $ZKP_{\text{MIX}}(,)$, we have $N' = N$, $sn'_j = sn_i$, $\mathbf{b'_j} = \mathbf{b_i}$ for any $i \in \{1, \cdots, N\}$ with π denoting a permutation from $\{1, \cdots, N\}$ to $\{1, \cdots, N\}$.

Deviation from these conditions would violate the mixnet security assumptions. Some digital ballots in $\mathcal{B}'$ are then privately decrypted and used for the RLA process. Let digital ballots after private decryption be denoted by $\mathcal{B}''$:

$$\mathcal{B}'' = \{(sn''_j, \mathbf{b''_j}) : j = \pi(i), i \in \{1, \cdots, N\}\}\}$$

The validity of $ZKP_{\text{DEC}}(,)$ ensures $sn''_j = sn'_j$ and $\mathbf{b''_j} = \mathbf{b'_j}$ (and hence $sn''_j = sn_i$ and $\mathbf{b''_j} = \mathbf{b_i}$), with $j = \pi(i)$. Otherwise, the security of the proof of decryption protocol would be violated.

Now let

$$\mathcal{B}''' = \{(sn_i, \mathbf{b_i}) : i \in \{1, \cdots, N\}\}\}$$

be the digital ballots recorded by the voting machine on the BB in Intermediate World 5. The tally in Intermediate World 5 ($\sum_{i=1}^{N} \mathbf{b_i}$) matches the result of decryption at step 6 of the protocol in the Intermediate World 6 ($D(\prod_{j=1}^{N} E(\mathbf{b'_j}))$). This equivalence is due to the homomorphic properties of the encryption algorithm. The private decryption at step 7 of the protocol in the Intermediate World 6 results in $\sum_{j=1}^{N} \mathbf{b'_j}$. Otherwise the security of the proof of decryption protocol would be violated. Since the paper path in both worlds is the same, the paper manifest in Intermediate World 5 can be selected to match that in Intermediate World 6. Each sequence of ballots in Intermediate World 6 corresponds to a sequence in Intermediate World 5 (established through the π^{-1} permutation function), maintaining identical discrepancy values. Consistency in both discrepancy values and tally across these intermediate worlds ensures an equivalent RLA risk.

Proposition 5 *Given that no "serial number error" is observed during the RLA (i.e., Section B.3, Verification at the Local Counting Center, item 7), for any adversary $\mathcal{A}$ who wins the Intermediate World 5 Game, there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 4 Game (or Intermediate World 3 Game for the Electronic-only model).*

Proof. In Intermediate World 5, when a digital ballot is chosen for the RLA, its discrepancy is set to the maximum possible value for that ballot unless a corresponding paper ballot with the same serial number and a lower discrepancy is identified in the paper record. If such a paper ballot exists in Intermediate World 5, it implies the coexistence of the same digital and paper ballot in Intermediate World 4. Consequently, the discrepancy for a digital ballot in Intermediate World 4 will not be less than its corresponding discrepancy value in Intermediate World 5.

Proposition 6 *Given that Eligibility Problem Stack includes all (and only) duplicate envelopes from the same voter (i.e., Section B.3, Verification at the Local Counting Center, item 2), Rejected Envelopes Stack includes all (and only) rejected envelopes (i.e., Section B.3, Verification at the Local Counting Center, item 3), for any adversary $\mathcal{A}$ who wins the Intermediate World 4 Game, then there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 3 Game.*

Proof. Let's start with the setup in the Intermediate World 4. Let N denote the total digital ballots on the BB. Assume the number of honest voters be denoted by n. Since there is a trusted party who honest voters can rely on to guarantee the presence of their ballots on the BB, n digital records on the BB correspond to these honest voters. So, we have $N \geq n$. In the Electronic&Drop model, although the envelopes mailed by honest voters might drop, the adversary is unable to stuff new envelopes on behalf of honest voters. Furthermore, the envelope that each honest voter mails, once received, remains unaltered. Therefore, paper ballots inside envelopes received from honest voters at the Local Counting Center in the Intermediate World 4, represented by $\mathcal{P}'_h$, form a subset of paper ballots viewed and mailed by the honest voters, represented by $\mathcal{P}_h$. Similarly, $\mathcal{P}_{\bar{h}}$ and $\mathcal{P}'_{\bar{h}}$ denote the paper ballots mailed by dishonest voters and received at the Local Counting Center, respectively.

Let the number of voters and the honest voters in the Intermediate World 3 be the same as in the Intermediate World 4 (i.e., N and n, respectively) with the same digital records recorded on the BB. Assume that the paper ballots provided to the voters in the Intermediate World 3 are identical to those in the Intermediate World 4 with one distinction: The serial numbers printed on $\mathcal{P}_h \backslash \mathcal{P}'_h$ and $\mathcal{P}_{\bar{h}} \backslash \mathcal{P}'_{\bar{h}}$ do not match any serial number on the BB. Figure 3 demonstrates these setups.

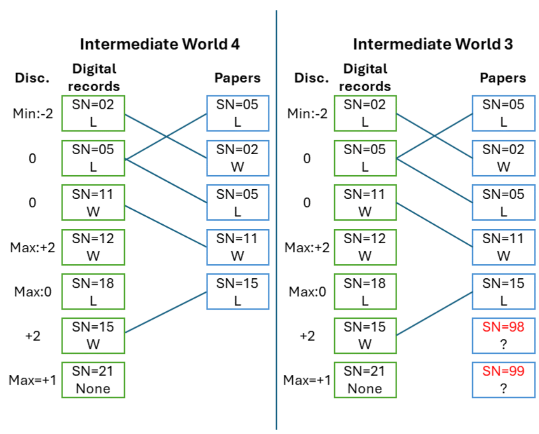

Fig. 3: Building ballots in Intermediate World 3 based on those in Intermediate World 4.

Then, these paper ballots do not match any records on the BB, and therefore, their presence in Intermediate World 3 does not affect any discrepancy values. Consequently, the discrepancy of each digital record in Intermediate World 3 will be equal to its corresponding ballot in Intermediate World 4. This shows for any adversary $\mathcal{A}$ who wins the Intermediate World 4 Game, there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 3 Game

Proposition 7 *For any adversary $\mathcal{A}$ who wins the Intermediate World 3 Game, then there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 2 Game.*

Proof. Let's start with the setup in Intermediate World 3. In this world, each digital ballot is categorized as either 'not matching', 'one-to-one', or 'duplicated'. We will prove that, in each of these categories, each ballot's discrepancy is maintained or decreased in the transition to Intermediate World 2. Figure 4 shows an example of the transition.

For each digital ballot in the 'duplicated' category, one of the paper ballots with a matching serial number that results in the minimum discrepancy is identified as the corresponding paper ballot for that digital ballot, and this correspondence is retained in Intermediate World 2. Remember that "minimum discrepancy" means "the minimum discrepancy number, where 0 is taken to be a match, overstatements are positive, and understatements are negative." Because we were choosing the "minimum discrepancy" for our RLA accounting in Intermediate World 3, it's obvious that these one-to-one matches we just made in Intermediate World 2 produces exactly the same discrepancies for every digital ballot in the 'duplicated' category.

In the example in Figure 4, we have a duplicate for $SN = 02$. The minimum discrepancy isn't the one that exactly matches, it's the 2-vote understatement (a "-2 discrepancy"). So in Intermediate World 2 we choose to keep the connection to the ballot with $(SN{=}02, W)$ (which therefore gets marked with a blue line and retained into Intermediate World 2). This therefore has the same discrepany in Intermediate worlds 2 and 3.

For digital ballots in the 'one-to-one' category in Intermediate World 3, there is no change in Intermediate World 2. In Figure 4, Serial numbers 05, 11 and 15 are in this category.

Finally, consider how to deal with digital ballots that are in the 'not matching' category in Intermediate World 3. We want to make a one-to-one correspondence with the ballot papers, of which we must (by assumption) have no more than we have electronic records. So we just make one up! We pick an arbitrary assignment of as-yet-not-matched digital records to as-yet-not-matched papers. Paper serial numbers are edited if the paper's serial number matched no digital record, or if it was a duplicate that was not selected as the minimum when dealing with the 'duplicated' category. Papers are added to make the number of digital and paper records equal, each with a serial number to match an as-yet-unmatched digital ballot.[10]

In Figure 4, this arbitrary assignment is performed for $SN = 12$ (which is matched to the unused duplicate of $SN = 02$), for $SN = 18$ (which is matched to the previously-unmatched $SN = 31$) and for $SN = 21$ (which is matched to an

[10] This is just a proof technique to demonstrate that the treatment of ballots is conservative. There is no actual alteration of serial numbers on ballots, nor addition of new ballot papers during the protocol.

invented ballot with an arbitrary vote selection). In each case, the serial number of the paper record is updated to match is corresponding digital one.

We need to argue that the RLA discrepancies must be no greater for these ballots, but that's easy because when we had no match (in Intermediate World 3) we assumed the worst-case discrepancy. Now we have a specific match for each of them, which can't, by definition, be worse than the worst case. (edited)

So, overall, this process produces a one-to-one match between digital and paper ballots, in which no digital record's discrepancy has increased from Intermediate World 3 to Intermediate World 2.

be denoted by $\mathcal{B}$ and $\mathcal{B}'$, respectively. Let the set of paper ballots and those corresponding to $\mathcal{B}'$ be denoted by $\mathcal{P}$ and $\mathcal{P}'$, respectively. The discrepancy for each ballot in $\mathcal{B} \backslash \mathcal{B}'$ is set to the maximum possible value for that ballot.

Now, let the number of voters in Intermediate World 2 be identical to that in Intermediate World 3. For each voter, the ballot selections on the paper and on the BB, and the serial number recorded on the BB in Intermediate World 2 are selected to be identical to those in Intermediate World 3. Additionally, the serial number on the paper ballot for each voter whose paper ballot in Intermediate World 3 is in $\mathcal{P}'$ would be the same in Intermediate World 2. However, the serial numbers on the paper ballots for other voters in Intermediate World 2 would be any arbitrary permutation of the serial numbers recorded on the BB for $\mathcal{B} \backslash \mathcal{B}'$ in Intermediate World 3. Consequently, discrepancies for ballots in $\mathcal{B}'$ would be identical in both worlds. But the discrepancy for each ballot in $\mathcal{B} \backslash \mathcal{B}'$ in Intermediate World 2 would be equal to or less than that in Intermediate World 3. Therefore, the RLA risk in Intermediate World 2 would be equal to or greater than in Intermediate World 3. Figure 4 demonstrates these setups.

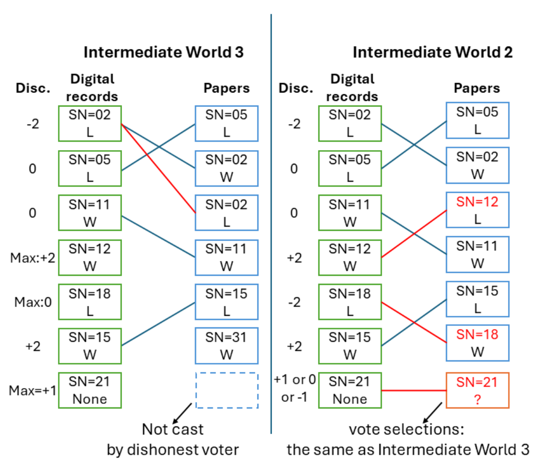

Fig. 4: Building ballots in Intermediate World 2 based on those in Intermediate World 3.

Proposition 8 *Given successful verification of the uniqueness of all serial numbers on the BB (i.e., Section B.3, BB transcript verification (public), item 5), for any adversary $\mathcal{A}$ who wins the Intermediate World 2 Game, then there exists an adversary $\mathcal{A}'$ who wins the Intermediate World 1 Game.*

Proof. Let the permutations that is applied to serial numbers before printing them on the paper ballots in the Intermediate World 2 be denoted by π. Let the paper ballot and the digital ballot for the voter i in the Intermediate World 2 be denoted by P_i and B_i, respectively. Now assume the Intermediate World 1 is established as follows. Although the paper ballot that is printed for the voter i is still P_i, the digital ballot that is recorded on the BB for the same voter would be B_j with $j = \pi^{-1}(i)$. This way discrepancies of all ballots in both worlds would be identical. Figure 5 demonstrates these setups.

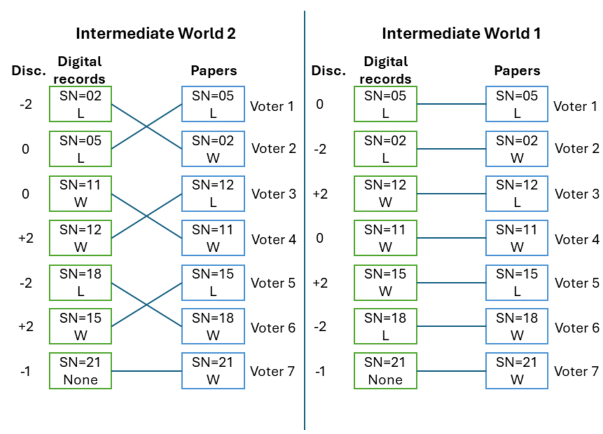

Fig. 5: Building ballots in Intermediate World 2 based on those in Intermediate World 3.

Proposition 9 *For any adversary $\mathcal{A}$ who wins the Intermediate World 1 Game, then there exists an adversary $\mathcal{A}'$ who wins the Ideal World Game.*

Proof. Due to the one-to-one correspondence between digital records and paper ballots (resulting from inclusion and exclusion criteria), and the fact that RLA involves the random selection of ballots, any sequence of paper ballots in the Ideal World corresponds to a matching sequence of digital ballots in the Intermediate World 1 with the same discrepancy values and hence with the same RLA risk.

C.2 Privacy

MERGE is intended to run as part of a larger protocol in which most ballots in the election are input by other means. However, privacy is difficult to model in that setting, so here we treat MERGE as a standalone protocol.

Our privacy proof therefore assumes an Electronic-only model—all paper ballots are guaranteed to arrive. It is the RLA step that allows for differencing attacks. Since the digital paths in both models are the same, the MERGE digital path alone protects privacy also.

Opening ballots for the RLA inevitably has some privacy implications, regardless of whether the ballots are mixed cryptographically or physically. MERGE neither introduces nor solves these problems. For example, if range voting is used with only two voters, then the attacker can distinguish a situation in which Bob votes 2 and Alice 0, from one in which Bob votes 1 and Alice 1, if either ballot is selected for audit, though the tallies are the same.

Even when no ballots are audited, small anonymity sets may reveal individuals' preferences, for example if there are only a very small number of remote votes sent to one Local Counting Center, and they all make the same choices. Group privacy of the whole set of remote voters may also cause concern, for example if the collective choices are strongly skewed relative to the rest of the population.

Table 1 summarizes what is visible to various participants. Access to both the voter's identifier and their vote is limited to the voting computer. The main privacy result is that the protocol does not reveal more about an individual voter's intent than the published mixed votes do.

Table 1: Data accessible to various entities

Entity	ID	SN	ID&SN	Vote	Tally
General public	✓	✓	×	×	✓[†]
Voting computer	✓	✓	✓	✓	✓
RLA observers	×	✓	×	✓	✓[†]
Postal worker	✓	×	×	×	✓[†]

[†]: visible on the BB

Definition 3. *The privacy attacker may view the following:*

1. *the complete contents of the BB, including vote tallies,*
2. *envelope stickers arriving at the Local Counting Center,*
3. *paper ballots that arrive at the Local Counting Center, after they have been physically disassociated from their identifying envelope,*
4. *decryptions (and proofs of proper decryption) of mixed votes that are sampled for audit,*

and may control:

1. *all voting computers for corrupted voters,*
2. *all but one of the mixing servers,*
3. *a sub-threshold number of decryption authorities.*

Subverting the privacy-preserving process of paper ballot handling (e.g. by opening the named envelope and reading its contents), is outside the attacker model. The privacy attacker may not open vote envelopes, and does not see honest voters' serial numbers or votes during the voting phase.

We use an instance of the privacy definition BPRIV from [6], in which the algorithms for Setup, Vote, Valid, Publish, Tally and Verify are derived from the MERGE protocol. Importantly, the *Tally* protocol is defined to publish both the decrypted vote tally from Step 7 and also the random permutation of individual votes from Step 9.

The attacker $\mathcal{A}$ is allowed to corrupt some voters, and is trying to guess how the honest ones voted. $\mathcal{A}$ is allowed to place different (encrypted) votes for each honest voter on BB_0 and BB_1. The challenger will choose $\beta \in \{0, 1\}$ at random. If $\beta = 0$, the adversary is shown the contents of BB_0, the tally from BB_0 and a validly-generated proof of mixing and decryption to produce that tally; if $\beta = 1$, the adversary is shown the contents of BB_1, *the tally from* BB_0 and a simulated proof that BB_1 produces that tally (which it may not do in reality). If the adversary cannot distinguish these two transcripts, it cannot guess how individuals voted.

Our proof relies upon privacy properties of both the underlying ElectionGuard cryptographic library and the mixing ZKP. In both cases, these assumptions seem reasonable though we are not aware of complete proofs in the literature.

Proposition 10 *MERGE in the Electronic-only model provides ballot privacy according to BPRIV [6], assuming that the underlying ElectionGuard ciphertexts satisfy NM-CPA and the mixing proof is zero knowledge.*

Proof This section sketches a proof of Proposition 10. We begin by restating the privacy definition BRPIV from [6]. We then list two important technical assumptions, which summarise the privacy provided by components. We then restate the proposition and provide the proof.

The setup assumes that all votes are opened for audit, which is the worst case. In practice, most typical runs of MERGE would be unlikely to open very many, if any, of the votes, thus revealing much less information than is assumed here.

Definition 4 (BPRIV, Adapted from [6] Definition 7). *Consider a voting scheme* $\mathcal{V} = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{Valid}, \mathsf{Publish}, \mathsf{Tally}, \mathsf{Verify})$ *for a set* I *of voter identities. The scheme has* ballot privacy *if there exists an algorithm* $\mathsf{SimProof}$ *such that no efficient adversary* $\mathcal{A}$ *can guess a randomly-chosen bit* β *with non-negligible advantage after playing the game* $\mathsf{Exp}^{\mathrm{BPRIV},\beta}_{\mathcal{A},\mathcal{V}}(\lambda)$ *defined below, where* λ *is the security parameter.*

- ***Setup.*** *The challenger picks a $\beta \in \{0,1\}$ uniformly at random. He sets up two empty bulletin boards BB_0 and BB_1 and two empty ballot boxes Box_0 and Box_1, creates a public key pair (pk, sk) and posts the public information pk on both boards. $\mathcal{A}$ is given access to either BB_0 and Box_0 if $\beta = 0$ or to BB_1 and Box_1 otherwise.*
- ***Voting.*** *$\mathcal{A}$ may make two types of queries.*
  - *($\mathcal{O}voteLR$) These model honest voters. $\mathcal{A}$ provides a voter identity id and two votes $(\mathbf{b_0}, \mathbf{b_1})$. The challenger generates a random serial number and creates a digital and paper ballot using $\mathbf{b_0}$. Similarly, the challenger generates another random serial number and creates a digital and paper ballot using $\mathbf{b_1}$. The challenger posts the digital ballot created from $\mathbf{b_0}$ and $\mathbf{b_1}$ on BB_0 and BB_1, respectively. She also drops the paper ballots created from $\mathbf{b_0}$ and $\mathbf{b_1}$ in Box_0 and Box_1, respectively.*
  - *($\mathcal{O}cast$) These are queries made on behalf of corrupt parties. $\mathcal{A}$ provides a vote $\mathbf{b}$. The challenger creates a random serial number and then creates a ballot and posts it on both BB_0 and BB_1. $\mathcal{A}$ is provided with two identical corresponding paper ballots and places either both or neither into their respective ballot boxes.*
- ***Tallying,*** *$\mathcal{O}\mathsf{Tally}()$. If $\beta = 0$, return valid tally $(r, \Pi) \leftarrow \mathsf{Tally}(BB_0, sk)$. If $\beta = 1$, set $(r, \Pi) \leftarrow \mathsf{Tally}(BB_0, sk)$ and return simulated tally $(r, \mathsf{SimProof}(BB_1, r))$. This includes showing $\mathcal{A}$ the stickers and paper ballots from Box_0.*

Assumptions MERGE relies on two critical technical components: the ElectionGuard protocol and the Verificatum mixnet. Proving that either of these have their required privacy properties is outside the scope of this paper. Instead, we state what we assume from them.

The first assumption is that the basic ElectionGuard encryption scheme, including its validity ZKPs, is non-malleable. This applies to the encryption of an individual ciphertext. We first recall the definition of non-malleability (NM-CPA).

Definition 5 (NM-CPA, from [6]). *NM-CPA is defined by the following game:*

1. *The challenger generates keys (sk, pk) and gives the public key pk to the adversary.*
2. *The adversary picks two messages m_0 and m_1 and hands them to the challenger, who chooses a random bit β and returns an encryption c^* of m_β.*
3. *The adversary may submit a vector $\boldsymbol{c}$ of query ciphertexts. For each query ciphertext $\boldsymbol{c}_i$, if $\boldsymbol{c}_i = c^*$, then the challenger returns $\perp$ (reuse of the challenge ciphertext is disallowed), otherwise he returns $\mathsf{Dec}(sk, \boldsymbol{c}_i)$.*
4. *The adversary submits a guess β' for β.*

The adversary's advantage is $|\Pr(\beta' = \beta) - 1/2|$. We say the encryption scheme is NM-CPA secure if this advantage is negligible in the security parameter.

This formalises the idea that the adversary cannot gain information about honest voters from reusing (possibly-manipulated versions of) their votes and thus inferring an honest vote from the tally. Together with the removal of duplicate ciphertexts, it implies that the votes of corrupted voters cannot depend on the votes of honest ones.

Assumption 11 *ElectionGuard's encryption scheme, including the validity ZKPs, is non-malleable (NM-CPA).*

As far as we know there is no proof of this, but non-malleability (NM-CPA) is proven for Helios's almost-identical encryption-with-ZKP scheme in [7].

Assumption 12 ([14]) *The Verificatum proof is an honest verifier zero knowledge proof of knowledge of the relation $\mathcal{R}_{\phi_{pk}} \vee \mathcal{R}_{\phi_{com}}$, where $\mathcal{R}_{\phi_{pk}}$ is the relation of valid shuffles and $\mathcal{R}_{\phi_{com}}$ is the relation that exposes the private commitment information. (See [14] for a precise statement.)*

Informally, this proof says "either I have performed a valid shuffle, or I have broken the commitment scheme." Assuming that the commitment scheme is sound, it implies a valid shuffle.

Assumption 13 *$\mathcal{A}$ cannot distinguish the following two situations:*

- *The $\beta = 0$ case: ballots arrive in signed envelopes and are physically shuffled, then the shuffled ballots are shown to the adversary.*
- *A different set of ballot papers arrive in the same signed envelopes, then a simulated shuffle is performed, with the same ballots from the $\beta = 0$ case shown to the adversary.*

The proof of privacy for MERGE employs a construction of SimProof that closely follows the proof in [6] for Helios, and adds a simulation of the shuffle proof guaranteed by Assumption 12.

The proof We are now ready to prove Proposition 10, which is restated here.

MERGE in the Electronic-only model provides ballot privacy according to BPRIV [6], assuming that the underlying ElectionGuard ciphertexts satisfy NM-CPA and the mixing proof is zero knowledge.

The definitions of the voting scheme $\mathcal{V} = (\mathsf{Setup}, \mathsf{Vote}, \mathsf{Valid}, \mathsf{Publish}, \mathsf{Tally}, \mathsf{Verify})$ are as follows. Setup Sets up the election public key and PKI for signatures.

$\mathsf{Vote}(id, v_0)$ Uses the election public key to encrypt the vote along with a randomly generated SN; signs it with the key associated with id (as specified in Voting step 3).

$\mathsf{Valid}(BB_\beta, b_\beta)$ Verifies the vote validity ZKPs and digital signatures for each vote, then removes any vote that shares a ciphertext with any other (as specified in Digital Processing Step 4).

$\mathsf{Publish}(BB_\beta)$ The challenger publishes the encrypted input votes (from either BB_0 or BB_1), and the corresponding ciphertexts output from the mix (under whatever random permutation π_0 or π_1 respectively the mix generates). Also the aggregated encrypted input votes (though this can be derived).

$\mathsf{Tally}(BB_0, sk)$ Returns the decrypted ballots in order from BB, accompanied by $ZKP_{\text{MIX}}(,)$ and the decryption ZKP $ZKP_{\text{DEC}}(,)$ for each vote.

According to BPRIV, if $\beta = 0$ the challenger directly posts the honestly-generated proof transcript $\mathsf{Tally}(\mathsf{BB}_0, \mathsf{sk})$; if $\beta = 1$ it instead posts the result r from $\mathsf{Tally}(\mathsf{BB}_0, \mathsf{sk})$ (which in our protocol consists of the aggregated tally and the permuted, decrypted ciphertexts) along with a simulation $\mathsf{Simproof}(\mathsf{BB}_1, \mathsf{r})$ of the rest of the transcript. We need to specify $\mathsf{Simproof}$ so that this simulated transcript is indistinguishable from the honestly-generated transcript from the $\beta = 0$ case.

$\mathsf{Simproof}(\mathsf{BB}_1, \mathsf{r})$ takes the true aggregated vote ciphertext from BB_1, the true mixing proof (input ciphertexts, output ciphertexts and $ZKP_{\text{MIX}}(,)$) from BB_1, and the decrypted votes from r, and uses the ZK simulator of the decryption ZKP to simulate both the decryptions of the aggregate (from Step 7) and the individual vote decryptions (from Step 9).

RLA. The adversary observes the RLA procedure run on paper ballots corresponding to the left board.

We model an adversary who colludes with sub-threshold number of guardians as having no knowledge of the private key.

We can now prove privacy in the interactive honest-verifier zero knowledge setting—the extension to the noninteractive setting can follow standard techniques used in [6].

Proof of Proposition 10

Proof. We need to prove that using SimProof for simulating the tally above (in the case $\beta = 1$) is indistinguishable from the real transcript (for the case $\beta = 0$). The proof is very similar to the proof of BPRIV for Helios in [6], except that the initial step simulates the mixing ZKP.

[Game G_{-1}] is the $\beta = 0$ case, where the challenger runs an honest decryption and tally of BB_0 and opens ballot box Box_0.

[Game G_0] is the same as G_{-1} except that all the ZKPs and the physical shuffle are simulated. The permuted list of decrypted output ballots, and the paper ballots shown to the adversary after shuffling, remain the same.

- Use Assumption 12 to produce a simulated shuffle ZKP indistinguishable from the real one.
- Use the Zero Knowledge property of ElectionGuard's decryption ZKPs to simulate them for all mixed votes.
- Use Assumption 13 to say that the physical shuffle can be "simulated" so that $\mathcal{A}$ cannot distinguish the real shuffle from the one in which votes were substituted.

By the Zero knowledge property, this is indistinguishable from Game G_{-1}.

The proof is now very similar to the proof of the equivalent step in [6]. Let n be the number of votes on BB_0. For $i = 1 \dots n$, define a series of games as follows.

[Game G_i] is derived from G_0 by replacing the first i ballots on BB_0 with the corresponding ballots from BB_1.

We now need to prove that, for $i = 1, \dots, n$ game G_{i-1} is indistinguishable from game G_i, from which it differs only in the substitution of ballot b_i on the BB.

Consider what the adversary could have used for b_i on board BB_1. If it is identical to b_i from BB_0 then distinguishing is impossible. If it is different, then either the adversary re-uses a ciphertext already on the board (in which case, the removal of duplicate ciphertexts corresponds to the return of $\perp$ in the NM-CPA game, no information is returned, and distinguishing is impossible), or the game is run and distinguishing the two boards—which differ only in one ballot—is equivalent to distinguishing those two ballots in the NM-CPA game. None of these options give $\mathcal{A}$ a better advantage than distinguishing the two ballots in the NM-CPA game, which we assume to be negligible (Assumption 11).

We rely on Assumption 13 to argue that the paper evidence does not help the adversary to distinguish these two games.

The last of these games (G_n) is exactly the $\beta = 1$ case, so we have shown that it is indistinguishable from the $\beta = 0$ game.

C.3 Receipt freeness

We assume the presence of a coercer outside the voting booth who is already aware of the target voter's identity and requests their serial number. Given that (1) serial numbers are ultimately disclosed on the BB and (2) serial numbers are sufficiently long, the voter is unable to provide false information about their serial number. Nevertheless, the coercer cannot rely on the voter's honesty regarding their cast vote. In other words, the voter cannot convince the coercer that the encrypted value $E(\mathbf{b_i})$ printed on the the sticker or published on the BB before the mixnet or the $E(\mathbf{b_i})$ in the Remote committed CVR (i.e. $(sn_i, E(\mathbf{b_i}))$ after the mixnet) is decrypted to the $\mathbf{b_i}$ that is claimed by the voter as her vote. However, as Table 2 shows, collusion between a coercer and auditors or observers at the Local Counting Center could enable the use of serial number data to connect voters' identities with their votes.

Appendix D.2 elaborates on the removal of serial numbers from our protocol. This ensures the preservation of voter privacy, even in the event of collusion between a coercer and auditors or observers at the Local Counting Center.

Table 2: Data accessible to the coercer and the RLA observers.

Entity	ID	SN	ID&SN	Vote	Tally
Coercer	✓	✓	✓	×	✓[†]
RLA observers	×	✓	×	✓	✓[†]

[†]: visible on the BB

D Using MERGE with other RLA styles and avoiding the publication of MERGE sub-tallies

D.1 VAULT-MERGE: incorporating MERGE in to an RLA that uses VAULT for all ballots

The VAULT RLA scheme [5] protects against pattern-based coercion attacks (often called "Italian attacks") by hiding individual ballots unless they are selected for RLA. This protects against both the problem of small sub-tallies for MERGE ballots and also the individual pattern-based coercion attacks on all ballots (at least, all ballots that are not selected for audit).

There are two main steps to incorporate MERGE into VAULT. In the first step, all the votes are tabulated on a VAULT BB. This step is simple: the MERGE ballots that are *output* by the mix are in exactly the right form for VAULT. The ordinary ballots from the Local Counting Center can simply be appended. The tallies are then computed over the whole set of votes.

In the second step, the RLA is conducted using the ballot-level comparison method, with samples taken at random from the VAULT-tabulated ballots. Discrepancies are calculated as follows:

– if the ciphertext is a MERGE ballot, it should be treated as described in section 4.4, using a ballot paper of matching serial number if possible, and a worst-case assumption if there is no ballot paper,
– if the ciphertext is an ordinary ballot, the paper ballot should be found and the discrepancy calculated as in a standard VAULT RLA.[11]

This method allows for a very efficient audit (ballot-level comparison for all ballots) with good privacy properties, even if the MERGE ballots are only a small set.

This would also work well if, for some reasons, the local electoral authority included some but not all ballots into a VAULT audit.

[11] The only technical restriction, compared with standard VAULT, is that for the MERGE ballots, the ciphertext must be decrypted rather than opened, because the authorities do not know the random factors used to generate it. The ordinary votes need to use the same encryption scheme as the MERGE ones, so that they can be aggregated together—they may be either decrypted like the MERGE ballots or opened (from the randomness used to generate them) like standard VAULT ballots.

D.2 Sub-MERGE: hiding MERGE ballots in larger batches

In this variant, serial numbers are not used, because paper ballots are not individually matched to their corresponding digital record. If it is known in advance that a certain ballot type will be processed using Sub-MERGE, the serial numbers can be omitted from ballots of that type. This has a significant advantage for defending against a coercer who colludes with people at the Local Counting Center, because even a voter who voluntarily reports their serial number (for example, by taking a photo) cannot be linked to their ballot. However, their anonymity set, among ballots of the same type, may still be small against a coercer who observes ballots in person at the Local Counting Center.

If serial numbers are omitted, it is probably convenient to add a ballot type indicator on the paper ballot—this information is present anyway, and having it printed explicitly is convenient for processing at the Local Counting Center.

Call the (small) set of MERGE ballots on the BB M. When $|M|$ is small, mixing does not have much benefit. We could use the ballots output from the mix, but it is just as valid to skip mixing and use the aggregate of M from *before* the mixing step that is published in Step 6 of the digital path.

The main idea is to make a larger batch that includes all the MERGE ballots and enough ordinary ballots to constitute a reasonable anonymity set of ballots of the same type. We tally the batch electronically, then use one of several possible methods for incorporating it into an existing audit. Table 3 presents an overview of the access levels granted to various stakeholders in Sub-MERGE. Plaintext sub-tallies from MERGE ballots are not calculated.[12]

Table 3: Data accessible to various entities in Sub-MERGE where SN is replaced with an explicit ballot type identifier.

Entity	Voter ID	Vote	MERGE Tally	Ballot type
General public	✓	×	×	✓
BMD machine	✓	✓	✓	✓
Coercer	✓	×	×	✓
RLA observers	×	✓	×	✓
Postal worker	✓	×	×	×

1. Gather a collection of local ballots L from those available in the Local Counting Center. They may have been cast in person, or they may be other (non-MERGE) absentee ballots. L should be chosen so that, when combined with the MERGE ballots, the anonymity set is large enough for the tallies to be published. Probably $|L \cup M| \approx 30$.

[12] They can of course be inferred by inspecting the arriving ballot papers, but this is not part of the privacy attacker model for this section.

We assume that ballots in L are already disassociated from the voter's name.
2. Encrypt the ballots in L using the encryption scheme from subsection B.2, including validity proofs. Post them on the BB.
3. Use the homomorphic property to compute the combined aggregate of $L \cup M$. Decrypt it and publish the ciphertext with proof of correct decryption on the BB. This gives us, on the BB:

$$E(\mathbf{Tally_{L \cup M}}), \mathbf{Tally_{L \cup M}}, ZKP_{\text{DEC}}(\mathbf{Tally_{L \cup M}}, E(\mathbf{Tally_{L \cup M}}))$$

The resulting data can be incorporated into a ballot-level comparison RLA by adopting the one-audit approach (see section D.2), or into a batch-level comparison RLA by simply tallying the batch manually (see below).

Batch-MERGE: batch-level comparison audits If the Local Counting Center is already using batch-level comparison audits, these batches can easily be incorporated. We can define the batches according to some physical convenience function—the corresponding electronic records should be easy to identify and need not be together on the BB. For instance, if batches are created based on the envelope arrival date, once a validly signed envelope is received, we place its ballot in the appropriate batch box and record the batch number on the BB.

This would make sense if the Local Counting Center already used batch-level comparison audits for their RLAs. The next audit steps would be as follows.

4. Manually tally the combined batch comprising L and the MERGE ballots.
5. Compute the overall discrepancy D between the electronic and manual tallies.

We now have the discrepancy between a plaintext electronic tally and a manual tally of paper ballots, each comprising both MERGE and ordinary ballots. This should be incorporated into the RLA statistics according to the rules for batch-level comparison RLAs, which should automatically deal appropriately with non-arrived MERGE ballots.

Ballot-level comparison RLA using ONEAudit If the Local Counting Center is already using ballot-level comparison audits, sub-MERGE batches can be efficiently incorporated using ONEAudit [13]. In this approach, discrepancies are computed by comparing ballots to their *overstatement net equivalent* CVRs (as defined below), rather than to individual CVRs.

Definition 6. *Two sets with the same number of CVRs are* overstatement net equivalent *if they produce the same totals.*

In this case, if a MERGE ballot is selected for audit, but no corresponding paper ballot has arrived, it is important to ensure the right worst-case assumption in the case that overstatements of $+2$ or more are possible. The next steps in the audit would be as follows.

4. Count the total number of MERGE electronic ballots in the batch, and subtract the number of MERGE ballots that have arrived, then explicitly add that number of "not arrived" zombie ballots to the catalogue of ballot papers.
5. The RLA then samples randomly from the catalogue of ballots, which includes:
   - batched MERGE paper ballots that have arrived (including any incorporated from the Local Counting Center),
   - zombies representing batched MERGE ballots that have not arrived,
   - ordinary paper ballots from the Local Counting Center, if there are any, (except the ones included in Batch-MERGE).

   It could also include any MERGE that were not included in sub-MERGE batches, which are sampled from the output of the mix as described in subsection 4.4—the different approaches can determined separately for different types of ballots.

If an ordinary paper ballot from the Local Counting Center (not in L) is sampled, it is dealt with however the Local Counting Center already deals with its ballot-comparison RLAs; if a (non-batched) MERGE ballot is sampled, it is matched according to serial number and audited as described in subsection 4.4. If a ballot in $L \cup M$ is sampled, use the discrepancy between its paper ballot and the *Overstatement Net Equivalent* CVR for $L \cup M$. If the ballot has not arrived, its "paper ballot" will be a zombie—account for it in the RLA as described in subsection 2.2.

Batching strategies and privacy implications It may even be possible to determine in advance which paper ballots will be in which batch—for example, if so few MERGE ballots are sent to one Local Counting Center that they are all treated as a single batch. Alternatively, batches could be assigned as the ballot papers arrive, based on their sticker.

Note that it does not matter whether the attacker knows, or can change, which ballots are contained in the same batch—this worst-case attack assumption is already part of the attacker model of batch RLAs. As long as the attacker does not know which batches will be sampled for audit, the risk limit is met.

Table 4 summarizes the access levels granted to various stakeholders in Batch-MERGE.

In the context of this table, we operate under the assumption that the batch number assigned to each participating voter is publicly disclosed on the BB. As indicated in the table, apart from the BMD machine, exclusive access to each ballot's vote content is granted only to the observers at the Local Counting Center. Nevertheless, the observers possess solely the batch number without any additional information to establish a connection between the vote contents and the respective voter's identity. Consequently, with a sufficiently large batch size, the privacy of the voter is preserved.

Table 4: Data accessible in Batch-MERGE.

Entity	ID	Batch No.	ID& Batch No.	Vote	Tally
General public	✓	✓	✓	×	✓[†]
BMD machine	✓	✓	✓	✓	✓
Coercer	✓	✓	✓	×	✓[†]
RLA observers	×	✓	×	✓	✓[†]
Postal worker	✓	✓	✓	×	✓[†]

[†]: visible on the BB

E Using MERGE with other RLA styles, avoiding the publication of MERGE sub-tallies

If a Local Counting Center has only a very small number of MERGE ballots, it may not be acceptable to publish their tally in item 7. When an attacker has direct access to the arriving mail ballot, we do not have a solution to this issue. However, if we consider a remote attacker who is looking at the sub-tallies on the BB, we can address the problem by including the MERGE tally into a larger tally of ballots. In more details, the ordinary ballots from the Local Counting Center can simply be appended to the MERGE ballots and then the tallies are computed over the whole set of votes. Appendix D describes multiple variants that obviate the need to publish the sub-tally of MERGE ballots in item 7 of the MERGE digital path. Some also avoid printing individual serial numbers on ballots, relying on batch numbers that identify a group of ballots instead. Appendix D.1 describes how to incorporate MERGE into audits using VAULT [5], retaining serial numbers, mixing, and individual ballot-level comparisons. Appendix D.2 describes how to incorporate MERGE ballots into larger batches and audit them as a batch—this requires neither mixing nor serial numbers, but may result in a less efficient audit.

These variations may also be useful for localities that use either VAULT or batch-level comparison audits anyway.

F Implementation results

In this section we discuss our prototype library for the MERGE protocol. The prototype library[13] has an ElectionGuard 2.0 implementation, as well as Elliptic Curve (NIST P-256) and mixnet implementations derived from the Verificatum library[14], along with an option to use GMP[15] for the low-level computations,

[13] Available at https://www.dropbox.com/scl/fi/4yzzpqrwwy49hjvzxm9as/Anonymous.rar?rlkey=gxaaefqv6emy7uxl6qst96zfu&st=4p3sr8gy&dl=0

[14] See https://www.verificatum.org/

[15] See https://gmplib.org/

also derived from Verificatum. For example, we have not (yet) implemented hash canonicalization as per the ElectionGuard specification. We also did not use a bullein board or run multiple instances of the mix, but both these things would be necessary for real operation.

For the measurements in this section we used a fast CPU (Intel Xeon CPU E5-2680 v3 @ 2.50GHz), for ballots with 7 contests, each having 4 selections. Each ElectionGuard contest has one more ciphertext than the number of real selections, so there are $\text{numberOfContests} + \text{totalNumberOfSelections} = 7 + (7 \times 4) = 35$ encryptions per ballot. Using elliptic curves, a single ballot like this can be encrypted (and the ZKPs generated) in about 20 ms. Encryption is done at the Remote Voting Center, which we expect to have modest hardware and be significantly slower. Ballots can be encrypted in parallel, which scales almost linearly with the number of available cores.

The rest of the cryptographic processing is done on a server, typically at each Local Counting Center. The cost of homomorphic tallying (.22 ms per encryption) and decryption of the tally (2.2 ms per encryption for 3 trustees) is probably negligible. The cost of ElectionGuard verification is about 15 ms per ballot before parallel speedup.

The largest time used in our workflow is for the mixnet and its verification. The election admin can run as many rounds of the mixnet as desired; each round consists of a shuffle/reencryption of the ballots and a generation of a proof of shuffle. Each round's proof must be verified, but only the last round's shuffle is used in the RLA. Using algorithms developed by Verificatum, one round of the mixnet takes about 60 msecs per ballot to run the mixnet and generate the proof. Verifying one round of the mixnet proof takes about 80 msecs per ballot. Both timings are single threaded on the fast CPU, and we get good speedup when running on multicore machines.

When doing ballot level RLA, the encrypted ballots selected for the audit must be decrypted using standard ElectionGuard threshold decryption. This costs about 2.3 msecs per encryption (including generating the decryption proof) for 3 trustees on the fast CPU, single threaded. Verifying the decryption proof costs about 15 ms per ballot.