A strongly software-independent voting system

***Abstract*—End-to-end verifiable (E2E-V) voting systems have been around for some time. However, their adoption is poor, largely because of electorates' discomfort with the underlying complex cryptography. Paper audit trail based systems have been effective in bringing easy-to-understand verifiability to electronic voting, but in most situations they need to trust the post-election custody chain of the paper audit trail.**

In this paper, we propose a novel polling booth voting protocol that demonstrates a one-to-one correspondence between the paper trail and the published cleartext electronic votes, without compromising individual or polling-booth level community vote secrecy. This not only brings cryptographic guarantees to the easily understandable paper audit trails, but also facilitates efficient and principled recovery from verification failures wherever possible. Our protocol is therefore strongly software independent.

Ours is a direct-recording electronic (DRE) protocol, but we do not require any voter-initiated cast-or-audit challenge typical of most E2E-V DRE systems. We can provide verifiable guarantee to each voter that votes are cast as intended. Our protocol is resistant to vote randomisation and chain voting type of coercion attacks. The protocol also does not reveal the vote to the DRE machine. The cognitive overhead for an average voter is minimal.

I. Introduction

As recent events have demonstrated, conducting large scale public elections in a dispute free manner is not an easy task. On the one hand, end-to-end verifiable (E2E-V) cryptographic voting systems can theoretically guarantee correctness of elections, but they are not yet very popular. Not only do they shift significant parts of the responsibilities of audit from the authorities to individual voters, but also observations such as the one made by the German Constitutional Court [1] makes depending entirely on cryptographic guarantees somewhat untenable:

> "The use of voting machines which electronically record the voters' votes and electronically ascertain the election result only meets the constitutional requirements if the essential steps of the voting and of the ascertainment of the result can be examined reliably and without any specialist knowledge of the subject ...
>
> The legislature is not prevented from using electronic voting machines in elections if the possibility of a reliable examination of correctness, which is constitutionally prescribed, is safeguarded. A complementary examination by the voter, by the electoral bodies or the general public is possible for example with electronic voting machines in which the votes are recorded in another way beside electronic storage."

On the other hand, there are systems that completely rely on paper-audit trails for correctness of elections [2], [3], [4], [5], [6], [7]. In such systems, reliable records of cleartext voter-marked paper ballots or voter-verified paper records (VVPRs) are maintained in addition to the electronic vote records. These systems use electronic counting primarily as an efficiency measure and perform rigorous statistical audits, called *risk-limiting audits* (RLAs), to establish that the winners declared by the electronic system match those that would have been declared by the full paper count. In case of serious mismatches, the electronic outcome is suggested to be replaced by the paper one.

However, paper records, which may be trustworthy at the time of voting, may not remain trustworthy at the time of counting or auditing. Hence RLA-based systems require the electorate to trust that a strict *compliance audit* [6], [7] of the custody chain of paper records – using traditional methods – is performed correctly.

It is thus natural to ask whether the strong cryptographic guarantees provided by E2E-V systems can be effectively combined with the understandability benefits of paper records, in a hybrid *dual voting* system [8]? However, to design such a scheme, one must be careful not to end up running two parallel and independent elections, only coupled loosely through simultaneously voting for both in the polling booth. If the two systems are not tightly coupled then it begs the question that which ought be the legal definition of the vote, and, in case of a mismatch in counts, which is the one that should be trusted and why? For the electronic record to be a valid proxy for the paper ballots, the electronic and the paper records need to be demonstrably in one-to-one correspondence. Moreover, in case of a mismatch in count in a tightly contested election, effective recovery would require identifying the errors to a few votes or polling booths, so that re-elections – if required – may be localised. This, in turn, is possible only if the two systems are tightly coupled and each can be used to audit the other.

In this paper we present a novel such dual-voting protocol with demonstrable one-to-one correspondence and strong recoverability properties. The electronic representation corresponding to each VVPR slip can be directly identified from a published list of plaintext votes, without allowing the corresponding voter receipt to be identified; and every voter receipt can be traced to the same published list of plaintext votes using a zero-knowledge protocol. In most cases mismatches between the paper and the electronic counts can be publicly narrowed down to the level of individual votes or problematic polling booths, without compromising individual or polling-booth level community vote secrecy. We believe that such transparency and fine-grained control is essential for public confidence in the electoral process.

Our second important contribution is the novel Direct-Recording Electronic (DRE) frontend of our protocol. In many situations a DRE protocol is preferred over hand-marked paper ballots in order to avoid the large number of 'invalid' ballots that may be produced by the latter, especially when the literacy levels are low. In most existing DRE-based E2E-V systems [9], [10], the cast-as-intended guarantee given to each voter is only partial. It typically depends on voters initiating a cast-or-audit challenge [11] and obtaining a proof that the *challenged* ballots were correct. There is no explicit guarantee on the finally cast vote because the machine's decision to cheat can depend on the incoming voter, and that, as such, is not sound. Such systems are susceptible to targeted attacks if it can be guessed that a given voter is not likely to challenge. Moreover, challenge-based systems are problematic in many cultures where challenging authorities or machines is not the norm. Our DRE protocol ensures that there is no possibility of such targeted attacks, and it does not require voters to initiate any challenge to obtain cast-as-intended guarantees.

Our protocol is also resistant to vote randomisation and chain voting types of coercion attacks. The DRE machines do not see the cleartext votes, thereby providing secrecy from any possibly compromised DRE machine which may leak information. Finally, the necessary steps to be performed by the voter (see Figure 3) are simple.

II. Our E2E-V requirements

In what follows we outline the security requirements for our E2E-V polling booth voting protocol. We consider first-past-the-post voting and discuss only the single race case.

A. System model

An E2E-V polling booth voting protocol has four main parties: voters, candidates (and polling agents who are their representatives), polling officers (POs), and the election authority (EA). Voters go to designated polling booths to vote and authenticate themselves to a PO. The PO allows them to vote after checking their eligibility. They then cast their votes in a private booth, either on a paper ballot or by pressing a button on a DRE machine. Voters are assumed to vote *bare-handed* [12], i.e., they do not carry any electronic devices. After voting completes, the EA collects vote records from all polling booths in the constituency, prepares data to be published on *public bulletin boards* [13], and announces the aggregate tally.

Anyone can verify from the published data that the announced tally is correct. Voters can individually verify that their votes are correctly recorded. During voting, the paper ballot or a VVPR may also get dropped in a box. In this case, the VVPRs from all the polling booths can be collected centrally to audit the published electronic data.

In Sections II-B to II-C we describe the security properties we need from an E2E-V system.

B. Verifiability

Democracy principles demand that it should not be necessary to trust any authorities, individually or collectively, for the correctness of the election process. Every component of the election process should be publicly verifiable.

Eligibility verification is outside the scope of this paper, hence trust on the polling officers for offline identity verification and eligibility checks – usually carried out in presence of the polling agents – is unavoidable. However, this trust must be publicly recorded, and we require the polling officers to certify each valid vote cast.

The overall correctness of voting is verified by the correctness of three steps: *cast-as-intended* indicating that the vote has been registered correctly, *recorded-as-cast* indicating the cast vote is correctly included in the final tally, and *counted-as-recorded* indicating that the final tally is correctly computed. *Recorded-as-intended* is a composition of the first two and *counted-as-cast* is a composition of the second two. *Spurious votes* are fake votes not certified by the presiding polling officer.

We capture the threat model for verifiability below:

Definition 1 (Threat model for verifiability)**.** *The adversary $\mathcal{A}_{\text{verifiability}}$ is a polynomially bounded adversary who may try to undetectably alter the outcome of the election. It*

1) *can corrupt the EA, the POs, the DRE machines, or any other authorities, except that it cannot ask the PO to allow ineligible voters to vote;*
2) *can alter or delete cast votes during polling, during the collection process, or while publishing on public bulletin boards;*
3) *can introduce fake votes in the system, i.e., those not certified by polling officers;*

Verifiability – as given Definitions 2-3 below – must hold in the presence of $\mathcal{A}_{\text{verifiability}}$.

Definition 2 (Universal verifiability)**.** *A voting system is* universally verifiable *if* anyone in the public *can verify using publicly posted data that each vote is recorded-as-cast and counted-as-recorded, and that there are no spurious votes in the final tally.*

Definition 3 (Individual verifiability)**.** *A voting system is* individually verifiable *if any voter can obtain a sound proof that their vote is recorded-as-intended in the final tally.*

A voting system is called *software-independent* if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome [14]. Software independence is a necessary condition for verifiability, because verifying the hardware-software subsystem of a DRE machine is almost always an intractable problem [15].

C. Secrecy and coercion resistance

Voting systems must never reveal how a given voter voted, to prevent vote selling and coercion.

In DRE systems where votes are directly recorded on a machine, it is typically required that the machine be trusted to not leak the voter's vote. Indeed, this is the setting under which we present a basic version of our protocol. However, the risk

of compromised DRE machines leaking votes through covert side-channels is one of the main reasons why DREs are not very popular. For the final version of our protocol, we consider the setting where the DRE does not learn the votes.

We assume that voters are bare-handed and they do not record their transaction with the DRE machine using any electronic means.

We also assume that any coercer can interact with the voter only before and after the voting process but not during it. This is reasonable because the presence of polling agents with the polling officer prevents opportunities for private conversations with a voter.

The custody chain of ballots from printing to the polling booths needs to be secured to prevent photocopying of ballot secrets. We assume that this can be ensured using standard tamper-evident paper covers or scratch surfaces [16].

Finally, the EA's backend infrastructure must be trusted to store the recorded votes and cryptographic secrets securely. Some E2E-V protocols attempt to distribute this trust by distributing the election secrets to multiple independent trustees. This, however, is somewhat artificial because there is usually a single election authority. Instead, solutions based on hardware enclaves and remote attestation [17] under regulatory oversight may be more suitable to mitigate this trust requirement (see Section VI-D).

We define the threat model for secrecy as below.

Definition 4 (Threat model for secrecy and coercion resistance)**.** *The adversary $\mathcal{A}_{\mathsf{secrecy}}$ is a polynomially bounded adversary who may try to learn others' votes, or coerce them to vote in a certain way, or be a voter itself and try to prove to others how it voted. It*

1) *can observe all voter receipts, VVPRs (during counting) and public outputs posted on bulletin boards;*
2) *can participate as a bare-handed voter;*
3) *can interact with other voters before and after the voting process but not during;*
4) *can control POs;*
5) *can corrupt DRE machines to reveal votes or other secrets;*
6) *cannot observe secrets of to-be-used paper ballots between printing to their usage without leaving a trace of tampering;*
7) *cannot corrupt the EA to reveal votes or other secrets.*

The following secrecy properties of Definitions 5-7 should ideally hold in the presence of $\mathcal{A}_{\mathsf{secrecy}}$.

Definition 5 (Individual vote secrecy)**.** *A voting system protects* individual vote secrecy *if given a (possibly malicious) voter's receipt and publicly posted data, no information can be derived about how the voter voted. That is, a voter cannot prove to anybody how she voted.*

Definition 6 (Community vote secrecy)**.** *A voting system protects* community vote secrecy *if given voters' receipts and publicly posted data, an adversary cannot determine how voters assigned to a given polling booth voted.*

Individual vote secrecy is a necessary condition for coercion resistance. Community vote secrecy is important to prevent profiling and targeting of voters belonging to a given locality. A polling booth typically admits a few thousand voters living in a neighbourhood, and revealing the vote tally of a polling booth compromises the neighbourhood. Hence, normally tally should only be published at an aggregate level for the entire constituency. Also, VVPR slips from multiple polling booths must be aggregated at a central place to protect polling booth-level voting patterns getting revealed during the VVPR audit.

Definition 7 (Coercion resistance [18])**.** *A voting system is* coercion resistant *if an adversary instructing a voter to vote in a certain way cannot determine whether she followed the specified instructions, given the receipt and publicly posted data.*

Randomisation attacks are a special case of coercion attacks where the adversary's aim is to force the voter to cast her vote randomly.

D. Recoverability

There should be a way to gracefully recover from elections that fail to verify either due to software bugs or interference of malicious actors. We define the threat model for recoverability as follows:

Definition 8 (Threat model for recoverability)**.** *The adversary $\mathcal{A}_{\mathsf{recoverability}}$ is a polynomially bounded adversary who may try to make election verifications fail. It*

1) *can do everything that $\mathcal{A}_{\mathsf{verifiability}}$ can do;*
2) *can participate as a voter, perhaps to claim falsely that her vote was not recorded-as-intended;*
3) *cannot mount denial-of-service attacks by interfering with the conduct of electoral processes or not supplying necessary information;*

The traditional definition of recoverability is called *strong software independence* which demands that *a detected change or error in an election outcome (due to a change or error in the software) can be corrected without re-running the election* [14]. However, this definition would almost certainly require depending entirely on the trustworthiness of the custody chain of VVPRs and falling back on manual counting. We modify this definition to imply that the verification failure can be narrowed down – wherever possible – to polling booth or even individual level granularity, so that, if absolutely required, the problem can be rectified by re-running the election locally before announcing results (Definition 9).

To achieve this, we require establishing a one-to-one correspondence of VVPRs with the electronic votes which enables identification of exactly those votes that do not verify. More sophisticated error recovery protocols can also be built using such correspondence.

Finally, dispute resolution is required to recover from malicious voters claiming that their vote was not recorded-as-intended (Definition 11).

The properties of Definitions 9-11 must hold in the presence of $\mathcal{A}_{\text{recoverability}}$.

Definition 9 (Strong software independence [14] - modified)**.** *A voting system is* strongly software independent *if a detected change or error in an election outcome (due to a change or error in the software) can be corrected without re-running the* entire *election, i.e., there is a way to narrow down the failures.*

Definition 10 (Public VVPR verifiability)**.** *A voting system has* public VVPR verifiability *if a one-to-one correspondence of the VVPR with the electronically recorded vote can be publicly demonstrated for any VVPR, without compromising individual and community vote secrecy (see Definitions 5 and 6).*

Definition 11 (Dispute resolution)**.** *A voting system has* dispute resolution *if an independent judge can correctly resolve disputes over whether a voter's vote was recorded-as-intended, by only consulting publicly posted data and without compromising the voter's individual vote secrecy (see Definition 5).*

It is to be noted that there is an inherent tension between strong software independence and community secrecy. Verification may fail with at least four types of cases – receipts or VVPRs with no corresponding electronic records, or electronic votes with no corresponding receipts or VVPRs – and these failures have to be accounted for. If during audit it turns out that the margin of votes between the winner and the next highest vote getter is too narrow to ignore the verification failures, then it may become imperative to identify the fouling polling booths (or subsets of individuals in them) and conduct polling again in them locally. However, this process can leak information. An adversary monitoring the published votes before and after repolling can determine which votes have changed and can correlate the changes with the polling booths involved in the repolling. This is unavoidable. However, such secrecy compromises should be limited to recovery procedures and should be minimised to the extent possible. In general, the larger the size of the repoll, the lesser is the leakage of information.

III. Existing E2E-V protocols and their limitations

In this section, we review some E2E-V voting techniques and protocols with respect to the requirements outlined above.

A. Frontends

1) Hand-marked paper ballot based frontends: There are several E2E-V voting systems that use optical scanning of hand-marked paper ballots. Some popular examples are Scratch & Vote [16], Punchscan [19], Prêt à voter [20], Scantegrity I-III [21], [22], [23]. The voters are required to mark their choices in the ballots, get them scanned and take home an encrypted part as receipt from which the votes cannot be determined. Post polling, copies of the encrypted receipts are displayed on public bulletin boards from where voters can verify that their votes have been recorded correctly. The ballot encryptions can be publicly verified by auditing a statistically significant random sample of ballots and disallowing audited ballots for actual voting.

2) DRE frontends and dispute resolution: Hand-marked paper ballots are dispute-free because voters use their own agency to mark the ballots. However, in DRE systems, when a voter transacts with a machine in the privacy of a polling booth without any witness, it is impossible to determine whether the voter's key-press was registered correctly [29]. In case of a dispute, allowing the voter to revote a few times is the only possibility.

Even if a voter accepts that the key-press is correctly recognised, a proof is required that the vote is recorded correctly internally. Protocols like STAR-vote [9] and Wombat [10], often categorised as 'encrypt-on-cast' protocols, print an encryption of the vote as a receipt and allow the voter to either cast the vote using this encryption or challenge the machine to open the encryption and demonstrate that it was correct. A statistically significant number of voters opting to challenge can ensure that the probability of detecting malfeasance is sufficiently high. However, this guarantee holds only if the decision to cheat is independent of the incoming voter. In some situations voters who are unlikely to challenge can be identified and selectively targeted; hence the protocol is not sound and there is no guarantee on the cast vote. Also, unless a voter pre-plans to challenge with a test vote, the challenge step may leak her voting intent to polling booth officials.

Markpledge [24] and Bingo voting [25] are DRE protocols that provide sound cast-as-intended guarantees, although perhaps at the expense of usability. Markpledge engages the voter in a human verifiable interactive zero-knowledge proof (ZKP) [26] requiring the voter to send her own random challenges. Bingo voting relies on the availability of a trusted random number generator (RNG) in the polling booth.

B. Randomization attacks

Many voting protocols that try to prevent the electronic recording machine – a scanner or a DRE – from learning the vote, do so by presenting a random permutation of the candidate order so that the voter's selection appears to be random and can only be decrypted at the backend. These are often susceptible to a coercion attack where a voter may be forced to vote for whichever candidate that appears at a fixed position in the permutation. The vote may then get randomized. Most systems that use Prêt à voter style ballots [20], [31], [16] are vulnerable to this attack.

C. Backends

Most E2E-V voting systems use either homomorphic tallying or mixnet-based [27] decryption of encrypted votes at the backend. In both backends, the encrypted votes are displayed on public bulletin boards which the voters can match with their receipts.

In homomorphic tallying based systems like Scratch & Vote [16] and STAR-Vote [9], the encrypted votes are added homomorphically to publicly obtain an encrypted tally of all

the candidates. The authorities decrypt the encrypted tally and publish a ZKP that the decryption was performed correctly.

In mixnet-based systems like Punchscan [19], Prêt à voter [20], Scantegrity I-III [21], [22], [23] and Markpledge [24], the encrypted votes available on the bulletin board are decrypted by a robust universally-verifiable backend mixnet that shuffles and decrypts cast ballots, then proves it did so correctly. However, in case of verification failures, it does not provide explicit mechanisms to localise the problem, and the entire tally has to be discarded.

D. Support for paper audit trails

In most hand-marked ballot schemes, such as Scratch & Vote [16], Punchscan [19] and Prêt à voter [20], the non-receipt part of the ballot is shredded to protect voter privacy. Thus, a paper record of the voter's voting intent cannot be maintained. Some systems such as Scantegrity I [21] and II [22] do maintain a paper audit trail but the paper record can be trivially linked with the voter receipt to reveal their vote; thus the paper audit cannot be performed publicly.

Methods that use homomorphic tallying cannot directly support a one-to-one correspondence with paper records since the cleartext votes corresponding to the encrypted votes are never revealed. Mixnet-based systems can support one-to-one correspondence with paper records only if an identifier can be attached to each output vote that can be matched with the paper record but not the voter's receipt.

Some protocols [8], [10], [30] do maintain paper audit trails. But they do so in a decoupled way such that VVPRs can be compared with the electronic votes only at the level of the final count. If the electronic counts are released at a coarse level of the entire constituency, it does not achieve strong software independence; if they are released at a fine level, say for each polling booth, then it directly compromises community vote secrecy.

A few systems do provide the one-to-one correspondence:

1) A variant of Prêt à voter [31]: This is a non-DRE protocol. A record of the voter's hand-marked choice is made using a carbon impression to create a paper copy of the marked ballot. Voter secrecy is maintained by ensuring that the serial number on the paper record cannot be matched with the serial number on the voter's receipt. The VVPR identifier needs to be decrypted during counting or audit to find the corresponding electronic record.

2) STAR-vote [9]: Although STAR-vote is based on a homomorphic tallying backend, it provides a one-to-one correspondence of paper records with electronic votes by injecting the encrypted votes to an independent parallel system consisting of a shuffler and decrypting to produces plaintext votes. This, however, defeats the very purpose of homomorphic encryption.

3) A protocol by Essex et al. [32]: This protocol fixes the non-public verifiability of Scantegrity II's paper audit by using a self-blanking invisible ink such that the paper record retains information about which vote was selected but without allowing linkage with voter receipts. This protocol is also a purely non-DRE system and requires voters to manually transcribe receipts. Also, it depends on specially designed invisible inks and pens which has its own verifiability and practicality concerns.

We give a detailed comparison of our protocol with other protocols supporting paper audit trails in Table I.

E. Risk-limiting audits

Risk-limiting audits (RLAs) verify that the winners declared by the electronic voting system match those that would be declared by a full hand count of the paper audit trail. The audit consists of repeated checking of paper ballots and continuing until either this fact is established beyond reasonable doubt or a full hand-count is performed to correct the winners reported by the electronic count.

All RLAs assume that the paper audit trail is secure. They have traditionally provided verifiability to non-E2E-V systems [2], [3], [4], [5]. For E2E-V systems, replacing the publicly verifiable electronic count with the paper count is questionable. In case of a count mismatch, it is the one-to-one correspondence between the paper vote and its electronic representation that allows a more nuanced approach to recoverability, by identifying failing votes and correcting them.

IV. Preliminaries and key cryptographic techniques

We let m and n denote the number of candidates and the number of voters participating in the election, respectively. A vote is thus an element from the set $\{0, 1, \ldots, m-1\}$, abbreviated as $[m]$. We define $\tilde{x} := x \mod m$ and $\hat{x} := x \text{ div } m$. We let i, j, k range over the set of voters, candidates and polling booths respectively (we drop the indices when the meaning is clear from the context). We denote public-key encryption and signature against entity X's public key by functions $E_X(\cdot)$ and $sign_X(\cdot)$ respectively.

G_1, G_2, G_T denote cyclic groups of prime order q ($q \gg m, n$) such that they admit an efficiently computable bilinear map $e : G_1 \times G_2 \to G_T$, i.e., for all $a, b \in \mathbb{Z}_q$ and generators g_1, g_2 of G_1 and G_2 respectively, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ and $e(g_1, g_2) \neq 1_{G_T}$, where 1_{G_T} denotes the identity element of G_T. We require that the n-Strong Diffie Hellman assumption [33] holds in (G_1, G_2) and that the discrete logarithm problem is hard in G_1. Throughout we use random generators $f_1, g_1, h_1 \in G_1$ and $f_2 \in G_2$.

A. Pedersen commitment

Given a message $\rho \in \mathbb{Z}_q$ we use the Pedersen commitment scheme [34], $C = g_1^{\rho} h_1^{r}$, where $r \in \mathbb{Z}_q$ is a secret randomness, to compute a value C that commits ρ.

Pedersen commitment is *perfectly hiding* because for each ρ there exists a unique r such that $C = g_1^{\rho} h_1^{r}$. The *computational binding* property, which demands that given a commitment C it must be hard to compute a different pair of message and randomness (ρ', r') with the same commitment, is derived from the hardness of the discrete logarithm problem, because distinct openings (ρ, r) and (ρ', r') of a given commitment

TABLE I
COMPARISON WITH SOME EXISTING E2E-V PROTOCOLS THAT SUPPORT PAPER AUDIT TRAILS.

Protocol	VVPR verifiability	Dispute resolution	Prevents randomisation attacks	DRE	Secrecy from the DRE / ballot scanners
Scantegrity II [22]	Non-public[1]	Yes	Yes	No	Yes
Sigma ballots [30]	Non-public[1], Aggregate[2]	Yes[3]	Yes	Yes	No
Benaloh's VOpScan [8]	Aggregate[2]	Partial[3,4]	Yes	No	No
Wombat [10]	Aggregate[2]	Partial[3,4]	Yes	Yes	No
Pret-a-Voter HRPAT [31]	Yes, but see Sec. III-D	Yes	No	No	Yes
STAR vote [9]	Yes, but see Sec. III-D	Partial[3,4]	Yes	Yes	No
Essex et al. [32]	Yes, but see Sec. III-D	Yes	Yes	No	Yes
Our protocol	**Yes**	**Yes**[3]	**Yes**	**Yes**	**Yes**

[1]The VVPR can be linked to the voter's receipt; thus this audit cannot be public.
[2]VVPR verification is only at an aggregate count level; VVPRs cannot be shown to be in one-to-one correspondence with the electronic votes.
[3]With the typical caveat for DRE systems that a bounded number of revotes are allowed in case of a dispute.
[4]There is no statistical guarantee about the correctness of the final cast ballot (not sound). Also, if the voter wishes to challenge the receipt, their voting intent may be leaked to polling officers.

$g_1^{\rho} h_1^{r} = g_1^{\rho'} h_1^{r'}$ reveal that $\log_{g_1}(h_1) = (\rho - \rho')/(r' - r) \mod q$.

Moreover, Pedersen commitment is additively homomorphic, i.e., if $C_1 = g_1^{\rho_1} h_1^{r_1}$ and $C_2 = g_1^{\rho_2} h_1^{r_2}$ are commitments of ρ_1 and ρ_2 respectively, then $C_1 C_2 = g_1^{\rho_1+\rho_2} h_1^{r_1+r_2}$ is a commitment of $\rho_1 + \rho_2$.

B. ZKP of set membership: C is a commitment of some $\rho \in \Phi$

We use the scheme proposed in [35] to provide a zero-knowledge proof that a given commitment C commits a message $\rho \in \Phi$, where Φ is a publicly available set. The main idea behind the scheme is that the verifier sends to the prover Boneh-Boyen signatures [33] on elements of set Φ under a fresh signing key. Then the prover can prove that C commits a member of the set by proving that it knows a signature on the value committed by C. If C does not commit a member of Φ then the proof fails because the prover does not know signatures on non-members of the set. The scheme is an honest-verifier ZKP of set membership if $|\Phi|$-Strong Diffie Hellman assumption holds in (G_1, G_2). Conversion to standard ZKP importantly requires the prover to verify all the verifier signatures. If proofs for multiple commitments are requested against the same set, the verifier's signatures can be reused and the proof for each commitment incurs an $O(1)$ online overhead.

See Appendix A for a more detailed description of the procedure and the associated security properties.

C. ZKP of reverse set membership: ρ is committed by some $C \in \Phi'$

In our protocol we also need a ZKP for the converse of above, i.e., to prove in zero knowledge that a message ρ is committed by some commitment $C \in \Phi'$, where Φ' is again publicly available. In what follows we present a novel scheme following the basic idea of above.

If we were to apply the same strategy as the previous ZKP, we would require signatures on commitments whereas the Boneh-Boyen signatures used earlier require messages to be in the exponent group $\mathbb{Z}_q$. Although a commitment can be hashed to an element in $\mathbb{Z}_q$, the correctness of this conversion would need to be proved via an expensive generic transformation of the hash function algorithm to arithmetic circuits.

Instead, we employ the following strategy. We use the BBS+ signature scheme [36] which allows one to present a commitment to the signer and obtain a *signature on the committed value* after proving to the signer that they know the committed value. Later, a proof of knowledge of the signature can be provided. Thus, we let the reverse set membership verifier act as the signer that sends signatures for each $C \in \Phi'$ after verifying that the prover knows the value committed by C. The prover thus obtains signatures on the values committed by each $C \in \Phi'$. To prove that a given message ρ is committed by some member of Φ', the prover gives a ZKP of knowledge of a signature on ρ. Since the prover only obtains signatures on values committed by members of Φ', it cannot prove successfully if no member of Φ' committed ρ. This protocol also enjoys the $O(1)$ online complexity as the previous protocol.

Using BBS+ signatures, we can prove security of the overall reverse set membership protocol under the $|\Phi'|$-Strong Diffie Hellman assumption in (G_1, G_2). See Appendix B for the detailed protocol and security proof.

V. THE BASIC PROTOCOL

For ease of exposition, we first present a basic version of our protocol, highlighting the main ideas behind the cast-as-intended guarantee at the DRE frontend and the VVPR one-to-one correspondence. In Section VI, we consider some essential extensions required to make the protocol satisfy all the requirements outlined in Section II.

A. Overview

1) Cast-as-intended guarantee at the DRE frontend: As highlighted in Section III-A2, the main challenge in a DRE frontend is providing sound cast-as-intended guarantees to bare-handed voters, without requiring them to initiate challenges. We achieve this by relying on pre-printed ballots and voters matching a small natural number ($< m$) on the DRE

receipt with a number printed against their chosen candidate on the ballot. For convenience, we call this pre-printed sheet a "ballot" even though it is not used as one in the basic protocol; we do use it as a traditional ballot from Section VI-A onwards.

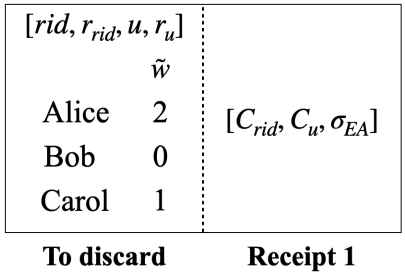

Fig. 1. Ballot design. Square brackets denote that the enclosed values are printed in a QR code.

Each ballot consists of a left half and a right half attached to each other through a perforated line (see Figure 1). The left half contains the candidate names in a canonical (e.g., alphabetical) order and a QR code containing ballot secrets. The QR code contains $rid \in \mathbb{Z}_q$, a random ballot identifier, $u \in \mathbb{Z}_q$, a randomly chosen one-time pad for masking the votes, and randomnesses r_{rid} and r_u for creating commitments $C_{rid} = g_1^{rid} h_1^{r_{rid}}$ and $C_u = g_1^u h_1^{r_u}$ on the right half. The masked value for each candidate $j \in [m]$ is given by $\tilde{w}_j = w_j \mod m$, where $w_j := u + j$, and is printed next to the candidate name in the left half. The right half also contains an EA signature on the (C_{rid}, C_u) tuple. The right half forms part one of the voter's take-home receipt.

We obtain cast-as-intended guarantees as follows. First, random ballot audits verify that commitment C_u to the one-time pad u and each masked value $\tilde{w}_j$ are constructed as specified. Unlike a voter-initiated challenge mechanism, random audits from a stack of pre-printed ballots is not vulnerable to any targeted attacks. During voting, the DRE produces part two of the voter's receipt containing a commitment C_v to the chosen vote v and the corresponding masked value $\tilde{w}_v$, along with w_v and a randomness $r_{w_v} = r_u + r_v$ defining a commitment to w_v. While in the voting booth, the voter verifies that the DRE's masked value $\tilde{w}_v$ matches the one printed next to her intended candidate on the ballot. This, together with the ballot audit and an at-home cryptographic check that $C_u C_v = g_1^{w_v} h_1^{r_{w_v}}$, assures the voter that the vote commitment C_v was indeed to her intended vote, by the additive homomorphism and the binding properties of Pedersen commitments.

2) One-to-one correspondence of VVPRs with electronic votes and recoverability: The DRE also prints a VVPR containing rid and human-readable v (or the candidate name). The voter can verify in the polling booth if the VVPR describes her vote correctly. The electronic vote-tabulation system consists of two bulletin boards: $\mathcal{BB}1$ containing vote commitments (C_{rid}, C_v) issued to voters and $\mathcal{BB}2$ containing plaintext votes (rid, v) unlinkable to the vote commitments on $\mathcal{BB}1$. Note that VVPRs can be matched with the plaintext votes on $\mathcal{BB}2$ but they cannot be matched with voter receipts, allowing public demonstration of a one-to-one correspondence of VVPRs with their electronic representations. In Section VI-B, we show that in case of a mismatch, the problem can be localised by identifying problematic polling booths from the VVPR.

To affix accountability of votes cast at a given polling booth, we require that the presiding polling officer at each polling booth must sign (C_{rid}, C_v) for every voter. These signatures appear next to the vote commitments on $\mathcal{BB}1$. The ZKP of Section IV-C allows proving for each plaintext vote on $\mathcal{BB}2$ that it was committed by some PO-certified vote commitment on $\mathcal{BB}1$ and thus is not spurious. Similarly, the ZKP of Section IV-B allows proving that the vote committed by each vote commitment on $\mathcal{BB}1$ was correctly included in $\mathcal{BB}2$. Any failing items can be individually identified and appropriate recovery action can be taken accordingly.

B. Detailed steps

1) Pre-polling preparation and ballot generation:

1) As one-time setup, random generators $f_1, g_1, h_1 \in G_1$ and $f_2 \in G_2$ are computed as outputs of a secure hash function on some public but unpredictable input (e.g., stock indices [37]). It will be assumed henceforth that nobody knows their mutual discrete logarithms.
2) The EA generates ballots as in Section V-A1 (see Figure 1). It ensures that the ballot identifiers rid are separated from each other by at least m. Each ballot is hidden under a tamper-evident sealed cover and a copy of C_{rid} is printed on the outside. Such covered ballots are sent to the polling booths. All official ballot commitments and signatures are published on a bulletin board $\mathcal{BB}0$.

2) Auditing ballots: At any point before, during or even after polling, random samples of the ballots kept at a polling booth can be audited by any concerned party. An audited ballot with the cover removed cannot be used for voting. Auditing a statistically significant sample of ballots gives a probabilistic guarantee that all ballots are correct.

Following needs to be checked during ballot audit (also see Figure 2):

1) commitments C_{rid} and C_u are constructed correctly as $C_{rid} = g_1^{rid} h_1^{r_{rid}}$ and $C_u = g_1^u h_1^{r_u}$ respectively;
2) for each candidate index $j \in [m]$, the masked value $\tilde{w}_j$ is correctly printed as $\tilde{w}_j = u + j \mod m$.

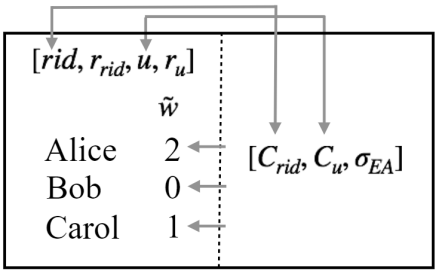

Fig. 2. Ballot audit. Gray arrows denote items to be checked during audit.

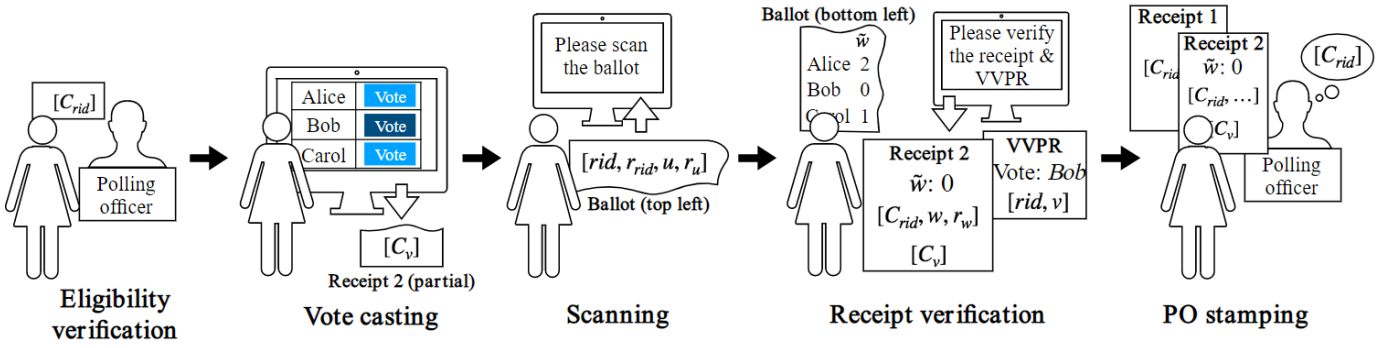

Fig. 3. The voter experience.

3) Voter experience: The voter experience in the polling booth is as follows (also see Figure 3):

1) **Eligibility verification:** The first step is eligibility verification at the PO's desk. After this, the PO allows the voter to pick a covered ballot from a stack and scans C_{rid} printed on the outside. The voter then proceeds to the DRE to cast her vote.
2) **Vote casting:** The voter casts her vote by pressing a button for her chosen candidate. The voter's choice is denoted by $v \in [m]$. At this point the DRE prints a commitment $C_v := g_1^v h_1^{r_v}$ to the voter's vote using fresh randomness $r_v \in \mathbb{Z}_q$ on a paper that we call the DRE receipt. The DRE receipt is not detachable yet.
3) **Scanning:** The voter opens the covered ballot and scans the QR code on the ballot left half, containing (rid, r_{rid}, u, r_u), to the DRE.
4) **Receipt verification:**
   a) The DRE computes $C_{rid} := g_1^{rid} h_1^{r_{rid}}$, $w_v := u + v$, $\tilde{w}_v := w_v \mod m$, $r_w := r_u + r_v$ and prints proof $(C_{rid}, w_v, \tilde{w}_v, r_w)$ on the DRE receipt, where $\tilde{w}_v$ is printed in a human-readable form.
   b) The DRE also prints a VVPR containing (rid, v) and in a human-readable form the name of the candidate at index v in the canonical candidate ordering.
   c) The voter checks if the masked value printed on the DRE receipt matches the masked value $\tilde{w}_v$ printed next to her chosen candidate on the ballot and also the candidate name printed on the VVPR. If satisfied, she presses accept. Otherwise, she asks for a revote.
   d) On the voter's acceptance, the VVPR drops to a VVPR box; the DRE prints a signature $\sigma_{DRE_k} := sign_{DRE_k}((C_v, C_{rid}, w_v, \tilde{w}_v, r_w))$ on the DRE receipt and allows it to now be detached.
   e) The voter takes the ballot receipt $R_1 := (C_{rid}, C_u, \sigma_{EA})$ and DRE receipt $R_2 := (C_v, C_{rid}, w_v, \tilde{w}_v, r_w, \sigma_{DRE_k})$ to the PO for stamping. (The left half of the ballot must be shredded.) $R := (R_1, R_2)$ forms the voter's take-home receipt.
5) **PO stamping:** The PO verifies if C_{rid}'s contained in both R_1 and R_2 match what he had scanned in step 1 and whether the DRE signature verifies. If yes, the PO internally stores a certificate $\sigma_{PO_k} := sign_{PO_k}((C_{rid}, C_v))$ and physically signs and stamps both the receipts R_1 and R_2, which the voter manually verifies.

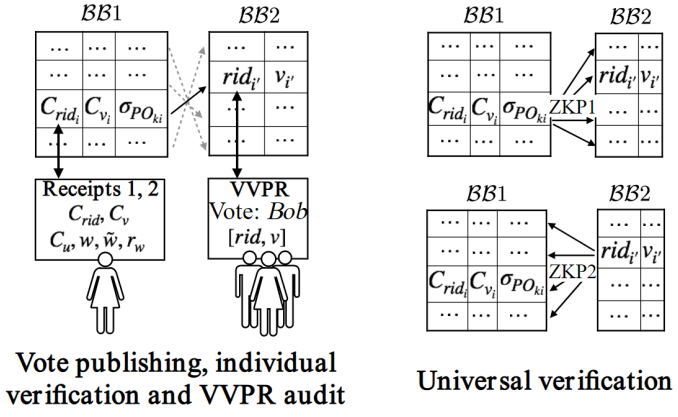

Fig. 4. Post-polling steps

4) Post-polling steps: Post-polling, the following steps are taken to publish and verify the results (also see Figure 4):

1) **Vote processing and publishing:**
   a) The DRE uploads $(C_{rid}, C_v, rid, v, r_{rid}, r_v)$ and the PO uploads certificates $(C_{rid}, C_v, \sigma_{PO_k})$ for each cast vote to the EA.
   b) The EA can match the corresponding DRE and PO records using C_{rid} and create internal records of the form $(C_{rid_i}, C_{v_i}, \sigma_{PO_{ki}}, rid_i, v_i, r_{rid_i}, r_{v_i})$. It can then publish a list containing records $(C_{rid_i}, C_{v_i}, \sigma_{PO_{ki}})$ on a bulletin board $\mathcal{BB}1$ and another list containing $(rid_{i'}, v_{i'})$ on another bulletin board $\mathcal{BB}2$, with $\mathcal{BB}1$ ordered by C_{rid} and $\mathcal{BB}2$ ordered by rid. It also publishes the final tally computed from $\mathcal{BB}2$.
   c) The EA also maintains a service to provide interactive ZKPs for proving the existence of a bijection between commitments (C_{rid_i}, C_{v_i}) posted on $\mathcal{BB}1$

and plaintext votes $(rid_{i'}, v_{i'})$ posted on $\mathcal{BB}2$ (see step 3 below).

2) **Individual verification:** The voter can check her receipt as follows:
   a) Check that $\tilde{w}_v = w_v \bmod m$ and $C_u C_v = g_1^{w_v} h_1^{r_w}$.
   b) Verify the ZKP of the validity of C_v as in step 3c and of membership of her vote on the final tally as in step 3d. (Optionally, a voter may only check whether an entry (C_{rid}, C_v) appears on bulletin board $\mathcal{BB}1$, thus relying solely on universal verification.)

3) **Universal verification:** Anyone can do the following:
   a) Verify that all v_i's on $\mathcal{BB}2$ are integers between 0 to $m-1$.
   b) Verify that all rid_i's on $\mathcal{BB}2$ are unique and apart from each other by at least m.
   c) **(ZKP0)** Perform interactive ZKP of Section IV-B for each C_{v_i} on $\mathcal{BB}1$ against set $[m]$ to verify that each C_{v_i} commits a valid vote.
   d) **(ZKP1)** Perform interactive ZKP of Section IV-B for each C_{rid_i} of $\mathcal{BB}1$ against set Φ denoting the set of rid's on $\mathcal{BB}2$; similarly for each $C_{rid_i} C_{v_i}$ against set Ψ denoting $rid_{i'} + v_{i'}$ for each row i' on $\mathcal{BB}2$.
   e) **(ZKP2)** Perform interactive ZKP of Section IV-C for each $rid_{i'}$ of $\mathcal{BB}2$ against set Φ' denoting the set of C_{rid}'s on $\mathcal{BB}1$; similarly for each $rid_{i'} + v_{i'}$ against set Ψ' denoting $C_{rid_i} C_{v_i}$ for each row i on $\mathcal{BB}1$.
   f) Verify that the number of entries in $\mathcal{BB}1$ and $\mathcal{BB}2$ are equal and the reported tally matches the one calculated from $\mathcal{BB}2$.

   For steps 3c-3e, the verifier sends signatures on set elements only once (see Sections IV-B and IV-C), to be reused by the EA for each row, resulting in an overall $O(N)$ complexity to verify all the votes.

4) **Public VVPR audit:** VVPRs from all the polling booths are collected at a central location and mixed to protect community vote secrecy during the VVPR audit. During this audit, anyone can demand to check that an electronic record corresponding to a VVPR containing (rid, v) exists on $\mathcal{BB}2$ and that the human-readable candidate name corresponds to v. *Note:* we do not obtain recoverability from VVPR mismatches until Section VI-B, because VVPRs do not yet contain polling booth information.

VI. Essential Extensions

We now discuss some essential extensions to the basic protocol to handle the following important issues: preventing DRE machines from learning votes, enabling recoverability by identification of problematic polling booths, mitigating vote randomisation attacks, and backend secrecy.

A. Preventing the DRE from learning the votes

A compromised DRE machine may leak information obtained from a voter to a coercer over covert side-channels; hence it should be prevented from obtaining cleartext votes. Also, since we reveal the cleartext votes corresponding to each rid on $\mathcal{BB}2$, the DRE should not learn the rid either. These requirements inherently conflict with VVPR printing because the VVPRs must contain both plaintext votes and the rid's for one-to-one correspondence. In what follows we present a modification to our basic protocol to achieve secrecy from the DRE. We assume for now that the EA is trusted for secrecy.

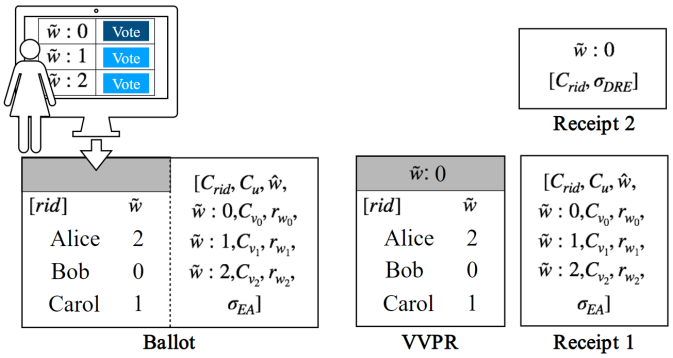

Fig. 5. Secrecy from the DRE machine. The machine sees only the gray region and prints $\tilde{w}$ in this region.

We suggest a variant of Figure 1's ballot design in Figure 5 such that the DRE machine does not obtain the secret u and does not need to produce C_v. Instead, the secrets u, r_u and rid, r_{rid} corresponding to each ballot are kept secure at the EA backend servers, and C_v and r_w corresponding to each $\tilde{w}$ are printed on the ballot right half. For ballot audits, the values u, r_u, rid, r_{rid} are obtained from the EA servers instead of the ballot left half. Ballot samples collected from polling booths without network facility have to be audited later.

The high-level protocol steps are as follows:

1) Instead of displaying candidate names, the DRE displays all possible values of $\tilde{w}$ and the voter selects the $\tilde{w}$ printed next to her preferred candidate on the ballot.
2) The voter separates the left half of the ballot and feeds only its gray region to the DRE. The DRE stamps the recorded $\tilde{w}$ in this region. This piece of paper forms the VVPR.
3) Next, the voter feeds the right half of the ballot to the DRE and obtains a signed DRE receipt containing C_{rid} and $\tilde{w}$. The voter needs to match the $\tilde{w}$ printed on both the top of the VVPR and on the DRE receipt with the $\tilde{w}$ printed next to her intended candidate on the ballot. As the voter leaves, the PO matches the C_{rid} on receipts 1 and 2, obtains C_v from receipt 1 for the $\tilde{w}$ printed on receipt 2, and signs (C_{rid}, C_v) as before.
4) The DRE uploads records of the form $(C_{rid}, \tilde{w}, C_{v_0}, ..., C_{v_{m-1}})$ to the EA; the PO uploads records of the form $(C_{rid}, C_v, \sigma_{PO})$ as before. The EA

can derive entries for both $\mathcal{BB}1$ and $\mathcal{BB}2$ from these records.

5) For individual verification one checks that $C_u C_v = g_1^{\hat{w}m+\tilde{w}} h_1^{r_w}$, where $\hat{w} := w \text{ div } m$, and C_v and r_w corresponding to the $\tilde{w}$ on the DRE receipt are looked up from the ballot receipt. Universal verification proceeds exactly as before. During VVPR audit, v can be obtained by looking up the candidate whose $\tilde{w}$ matches the $\tilde{w}$ printed at the top of the VVPR. Then, (rid, v) can be verified against $\mathcal{BB}2$.

Since the DRE only obtains $\tilde{w}$ and the public receipt component of the ballot, it does not learn anything about the vote. Note that the individual verification check is identical as before because $w = \hat{w}m + \tilde{w}$.

B. Enabling recoverability by adding polling-booth information to VVPRs

We suggest the following extension to the above protocol to add the polling booth information to the VVPR to allow recoverability while protecting community-level vote secrecy.

To the left half in the non-gray region of each ballot we add $C_k = g_1^k h_1^{r_k}$, a commitment to the polling booth identifier k, with the values k, r_k held in secret by the EA. The ballot audits can also verify that C_k opens to the correct polling booth identifier. Then, if a mismatch happens during VVPR audit, the EA is asked to open the C_k printed on the VVPR.

With this extension, we show a public recovery protocol from potential failures during universal verification or VVPR audit in Table II.

In case the recovery protocol requires a local re-election at a polling booth, the tally of the rest of the polling booths can be publicly verified by asking the EA to remove vote commitments from the offending polling booth on $\mathcal{BB}1$ and also the corresponding plaintext votes on $\mathcal{BB}2$ and performing the universal verification steps again.

C. Prevention of vote randomisation attacks

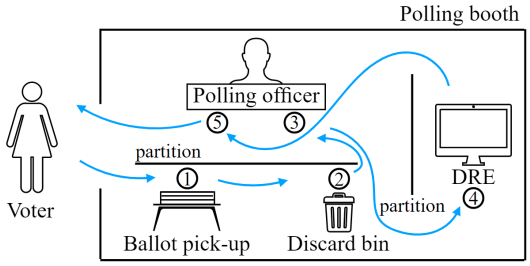

Fig. 6. Modified voter workflow for preventing randomisation attacks. 1) Ballot pick-up; 2) Discarding the detachable ballot type tag; 3) Eligibility verification; 4) Vote casting; 5) PO stamping.

The protocol described so far is actually susceptible to the attack where a coercer may force a voter to vote randomly by asking for a particular masked value $\tilde{w}$ on the DRE receipt. If the voter's preferred candidate does not have the required $\tilde{w}$, her vote would be randomised.

To mitigate this attack, we allow a coerced voter to select a ballot type such that the masked value corresponding to the voter's preferred candidate matches the one demanded by the coercer. Specifically, we require that covered ballots be kept inside a private room and the ballot type, essentially $\tilde{u} := u \mod m$, be printed outside the cover as a detachable component. Also, the C_{rid} printed on the outside of the cover must itself be covered by a tamper-evident mechanism because the ballot type $\tilde{u}$ is now publicly visible and we cannot leak $\tilde{u}$ corresponding to a C_{rid}.

The following extra steps need to be taken (see also Figure 6):

1) Voters select their desired ballot type in private, detach this information and then proceed to the PO for eligibility verification. The PO uncovers C_{rid} and scans it.
2) Ballot audits check that the ballot type printed outside the cover matches the actual candidate ordering in the ballot inside.

Candidate look-up charts for each ballot type can be displayed in the private room to help coerced voters to pick the correct ballot type. Non-coerced voters do not incur any additional cognitive overhead and they can simply pick any ballot.

D. Mitigating secrecy leak risks at the EA backend

Existing approaches to mitigate the risks at the EA backend generally rely on techniques like threshold encryption to distribute the trust to multiple independent trustees. However, in practice, in most jurisdictions there is only a single election authority and such distribution of trust is somewhat artificial. An alternative would be to rely on isolated execution environments called hardware *enclaves* [17] that can run programs that are pre-approved and publicly announced by an independent regulator. Such enclaves protect the confidentiality and integrity of the program memory from even privileged insiders, and can be verified by regulators through remote attestation.

All ballots must carry a digital signature that attests that the printed commitments were produced by an approved enclaved program, which the ballot audits must verify. Also, only such enclaved programs may decrypt the cast votes. Note, however, that even with such enclaves, one needs to trust the processes at the EA for ballot printing to maintain secrecy. These processes must also be under regulatory oversight to ensure that no large scale electronic recording is possible.

VII. Security Analysis

We show that the full protocol, with the extensions of Section VI, achieves the security requirements of Section II.

A. Universal and individual verifiability

We start by defining *cast votes* for the verifiability definitions to be vote commitments that appear on $\mathcal{BB}1$.

Theorem 1. *The protocol is universally verifiable (Definition 2) in the presence of adversary $\mathcal{A}_{\mathsf{verifiability}}$ (Definition 1).*

TABLE II
DIAGNOSIS AND A PUBLIC RECOVERY PROTOCOL — TO BE EXECUTED SEQUENTIALLY — TO HANDLE FAILURES IN UNIVERSAL VERIFICATION (**UF**) AND VVPR AUDIT (**VF**).

Failure mode	**Diagnosis**	**Recovery action**
(UF0) The EA obtained a vote commitment (C_{rid}, C_v) not generated by it in step 2 of Sec. V-B1.	Some PO used an unofficial ballot.	EA shows that $(C_{rid}, C_v) \notin \mathcal{BB}0$, flags the polling booth and invalidates this row on $\mathcal{BB}1$. Subsequent to this, the EA is accountable for any $v \notin [m]$ or rid's clashing within $\pm m$ on $\mathcal{BB}2$, or ZKP0 failing for any C_v on $\mathcal{BB}1$.
(UF1) ZKP1 fails for some rows on $\mathcal{BB}1$ and ZKP2 fails for some rows on $\mathcal{BB}2$.	ZKP1 may fail because the EA obtained an unofficial vote commitment; ZKP2 may fail because of a spurious vote injection.	For ZKP1 failures, the offending polling booths for the failing rows are publicly identified from $\mathcal{BB}1$, after the EA has proved that it did not generate them in UF0. For ZKP2 failures, EA is accountable.
(UF2) Size of $\mathcal{BB}1$ does not match with the size of $\mathcal{BB}2$.	Multiple vote commitments on $\mathcal{BB}1$ open to the same plaintext vote on $\mathcal{BB}2$, due to malfeasance at the EA. (If a PO provided clashing C_{rid}'s, an honest EA would create corresponding rid's on $\mathcal{BB}2$ and flag it during UF0.)	EA is asked to invalidate all duplicate entries from $\mathcal{BB}1$ until its size matches that of $\mathcal{BB}2$.
(VF0) No row on $\mathcal{BB}2$ exists for an rid on a VVPR.	Either the VVPR is spurious or a cast vote is lost.	EA is asked to open C_k from the failing VVPRs to identify the offending polling booth.
(VF1) No VVPR exists for an rid on $\mathcal{BB}2$. (Special methods may be required to detect this failure mode.)	The VVPR was lost but the PO had certified the corresponding commitment.	Offending polling booths can be publicly identified by running ZKP1 against a set of plaintext votes that excludes such entries.
(VF2) Votes on some VVPRs do not match with the corresponding rows on $\mathcal{BB}2$ containing the same rid.	This indicates clear malfeasance either at the VVPR or the electronic vote.	EA is asked to open C_k from the failing VVPRs to identify the offending polling booth.
(VF3) Multiple VVPRs match with the same entry on $\mathcal{BB}2$ or one VVPR matches to multiple entries on $\mathcal{BB}2$.	In the first case, the VVPR is clearly maliciously created. Second case not possible because rid's on $\mathcal{BB}2$ are separated by at least m.	EA is asked to open C_k from the failing VVPRs to identify the offending polling booth.

Proof. A bijection between vote commitments on $\mathcal{BB}1$ and the plaintext votes on $\mathcal{BB}2$ directly leading to the final tally directly establishes the *counted-as-cast* component of *universal verifiability*. Lemmas 1 - 3, and that because of the binding property a single vote commitment in $\mathcal{BB}1$ cannot correspond to multiple rows in $\mathcal{BB}2$, establishes the bijection.

Lemma 2, and that each committed vote in $\mathcal{BB}1$ is digitally signed by a polling officer ensures that there can be *no spurious vote injection* .

We say that a row (C_{rid}, C_v) in $\mathcal{BB}1$ *commits a row* (rid, v) in $\mathcal{BB}2$ if C_{rid} commits rid and C_v commits v.

Lemma 1. *Each row in $\mathcal{BB}1$ commits some row in $\mathcal{BB}2$.*

Proof. By the soundness of the set membership proofs in ZKP1, for each row (C_{rid}, C_v) on $\mathcal{BB}1$ *a)* C_{rid} commits $rid_{i'}$ for some row i' on $\mathcal{BB}2$, and *b)* $C_{rid}C_v$ commits $rid_{i''} + v_{i''}$ for some (possibly different) row i'' on $\mathcal{BB}2$.

Dividing the two commitments and using the binding property of Pedersen commitments, we get $rid_{i''} - rid_{i'} = v - v_{i''}$, where v denotes the value committed by C_v. By the soundness of ZKP0 for C_v, $v \in [m]$; by step 3a of Section V-B4, $v_{i''} \in [m]$; thus $-m < v - v_{i''} < m$. Then, $i' = i''$, implying that they must be from the same row, because by step 3b, for any two distinct rows of $\mathcal{BB}2$, the rid's must be separated by at least m. □

Lemma 2. *Each row in $\mathcal{BB}2$ is committed by some row in $\mathcal{BB}1$.*

Proof. By the soundness of the set membership proof in ZKP2, for each row (rid, v) on $\mathcal{BB}2$, rid is committed by some $C_{rid_{i'}}$ on $\mathcal{BB}1$.

By Lemma 1, there exists a row (rid', v') on $\mathcal{BB}2$ such that $C_{rid_{i'}}$ commits rid' and $C_{v'_i}$ commits v'. Since $C_{rid_{i'}}$ commits rid and distinct rows on $\mathcal{BB}2$ cannot have the same rid (by step 3b of Section V-B4), $C_{v'_i}$ must commit $v = v'$. □

Lemma 3. *Each row in $\mathcal{BB}1$ commits a distinct row in $\mathcal{BB}2$.*

Proof. If two rows in $\mathcal{BB}1$ commit the same row in $\mathcal{BB}2$ then some row in $\mathcal{BB}2$ must be committed by no row in $\mathcal{BB}1$ (because the sizes of $\mathcal{BB}1$ and $\mathcal{BB}2$ are equal and no row in $\mathcal{BB}1$ can commit two distinct rows in $\mathcal{BB}2$ due to the binding property). By Lemma 2, this is not possible. □

□

Theorem 2. *The protocol is individually verifiable (Definition 3) in the presence of adversary $\mathcal{A}_{\mathsf{verifiability}}$.*

Proof. We assume that the ballots are well formed (as may be ascertained by auditing a statistically significant sample of ballots) and the individual verification steps 2 of Section V-B4 for the voter passed.

Lemma 4 below guarantees that the commitment C_v on the voter's receipt is a commitment to her intended vote. If an entry corresponding to (C_{rid}, C_v) in the voter's receipt appears on $\mathcal{BB}1$, this gives a direct cast-as-intended guarantee to the voter. Also, the steps for ZKP1 using commitments (C_{rid}, C_v) give her a direct recorded-as-intended guarantee, by Lemma 1. Thus, the protocol is *individually verifiable*.

Lemma 4. *The commitment C_v on the voter's receipt is a commitment to the voter's intended vote v.*

Proof. Suppose the adversary knows $v^*, r_{v^*} \in \mathbb{Z}_q$ such that $C_v = g_1^{v^*} h_1^{r_{v^*}}$. We show that $v^* = v$.

First, recall that if the ballot is well-formed then $\tilde{w}_j = u + j \mod m$ for each candidate index $j \in [m]$. Since $\tilde{w}$ on the DRE receipt is matched by the voter against her intended candidate's $\tilde{w}$ on her ballot, we have $\tilde{w} = u + v \mod m$. Also, there exists r_u such that $C_u := g_1^u h_1^{r_u}$.

It is checked in Section VI-A that $C_u C_v = g_1^{\hat{w}m+\tilde{w}} h_1^{r_w}$, where C_v and r_w are on the ballot receipt at row $\tilde{w}$ printed on the DRE receipt, and $\hat{w}$ is on the ballot receipt. Thus, $g_1^{u+v^*} h_1^{r_u+r_{v^*}} = g_1^{\hat{w}m+\tilde{w}} h_1^{r_w}$. By the computational binding property of Pedersen commitments, we have $(u+v^*) = \hat{w}m + \tilde{w}$.

Since $\tilde{w} = u + v \mod m$, $\tilde{w} = \tilde{u} + v - bm$ where $b = 0$ if $\tilde{u} + v < m$ else $b = 1$. Thus:

$$\begin{aligned} \hat{u}m + \tilde{u} + v^* &= \hat{w}m + \tilde{u} + v - bm \\ \therefore \quad \hat{u}m + v^* &= (\hat{w} - b)m + v \end{aligned} \tag{1}$$

Since $v \in [m]$ and $v^* \in [m]$ (because of the soundness of ZKP0 on C_v on $\mathcal{BB}1$), we must have $v^* = v$. □

□

B. Individual/community vote secrecy and coercion resistance

Theorem 3. *The protocol's public outputs preserve individual and community vote secrecy (Definitions 5 and 6) in the presence of adversary* $\mathcal{A}_{\mathsf{secrecy}}$.

Proof. By Lemma 5 below, no voter's receipt leaks any information about their votes. Also, the voter receipts cannot be linked with plaintext votes on $\mathcal{BB}2$.

Entries in $\mathcal{BB}1$ and $\mathcal{BB}2$ are unlinkable to each other because of their independent ordering and the hiding property of commitments. The ZKPs of set membership each preserve unlinkability. $\mathcal{BB}2$, which displays votes in clear text, does not contain any polling booth level identifiers and $\mathcal{BB}1$, which contains polling booth information, does not reveal any information about any vote.

By Theorem 6, VVPRs cannot be linked with voters' receipts and they also protect polling booth information.

Lemma 5. *The voter's receipt does not leak any information about her cast vote v, even if the DRE machine or the voter is malicious.*

Proof. The only information revealed by a covered ballot before polling is its ballot type $\tilde{u}$, and when the voter detaches $\tilde{u}$ before coming to the PO, this information is lost and all ballot types look identical to all (see Section VI-C).

As per Figure 5, the DRE receipt contains $(C_{rid}, \tilde{w}, \sigma_{DRE_k})$ and the ballot receipt contains $(C_{rid}, C_u, \hat{w}, C_{v_0}, r_{w_0}, ..., C_{v_{m-1}}, r_{w_{m-1}}, \sigma_{EA})$. The commitments do not leak any information about the committed values because of their hiding property; r_{w_j} perfectly blinds secrets r_u and r_{v_j} corresponding to commitments C_u and C_{v_j}. Value $w := \hat{w}m + \tilde{w}$ masks the vote because as long as w falls in the interval $[m-1, q-1]$, for each $v \in [m]$ there exists a unique secret $u \in \mathbb{Z}_q$ such that $u + v = w$, and the probability of w falling outside this range for uniformly distributed $u \in \mathbb{Z}_q$ is negligible since $q \gg m$. Finally the signatures do not reveal anything the corresponding messages do not.

Since the DRE machine only obtains the ballot receipt and $\tilde{w}$ derived from the masked value w, it does not learn any information about the vote. Also, bare-handed voters cannot disclose rid. □

□

Theorem 4. *The protocol is coercion-resistant (Definition 7) in the presence of adversary* $\mathcal{A}_{\mathsf{secrecy}}$.

Proof. $\mathcal{A}_{\mathsf{secrecy}}$ acting as a coercer can specify its instructions only before voting (Definition 4). Asking the voter to vote using an opened ballot, as in "chain voting", cannot be fulfilled because the PO matches the C_{rid} on the outside of the sealed cover to that produced by the voter. Also any instructions involving the non-human-readable parts of the receipt cannot be fulfilled by a bare-handed voter.

Thus, instructions can only be on $\tilde{w}$ (i.e., randomisation attacks). However, for any $\tilde{w}$ insisted on by the coercer, the voter can vote for any v^* by choosing a ballot type $\tilde{u} := \tilde{w} - v^* \mod m$ (in practice, with the help of look-up charts). We assume that ballots are well formed as may be ascertained by prior audit of a random sample of ballots. Theorem 3 then guarantees that it is impossible for the coercer to determine if the voter followed the instruction or not. □

C. Dispute resolution, VVPR verifiability and strong software independence

Theorem 5. *If the voter accepts the DRE receipt in step 4c of Section V-B3, the protocol has dispute resolution (Definition 11) in the presence of adversary* $\mathcal{A}_{\mathsf{recoverability}}$ *(Definition 8).*

Proof. We must disambiguate between honest voters' complaints and $\mathcal{A}_{\mathsf{recoverability}}$ raising false disputes. We construct a public algorithm J for dispute resolution. It takes as input a receipt R^* produced by a voter and publicly posted data and proceeds as follows:

1) If the physical PO signature in receipt R^* fails, decide against the voter.
2) Otherwise, decide in favour of the voter if and only if individual verification steps fail with receipt R^*.

Lemma 6 directly implies the theorem statement.

Lemma 6. *Algorithm J decides in favour of the voter if the voter is honest, and against the voter if the EA, PO and the DRE machine are honest.*

Proof. Case 1 (The voter is honest): The physical PO signature is manually verified by the voter in step 5 of Section V-B3, and is stamped in the presence of polling agents; so an honest voter cannot produce a receipt with invalid signatures. If the vote is not recorded as intended, then by Theorem 2, the individual verification must fail, and J decides in favour of the voter.

Case 2 (The EA, PO and the DRE machine are honest): In this case, if the physical signature fails then J always decides against the voter. Else, the PO signature and matching of C_{rid} (step 5 of Section V-B4) implies that the produced receipt's (C_{rid}, C_v) is correctly recorded in the cast vote list on $\mathcal{BB}1$. By the completeness of set-membership proofs in ZKP0/ZKP1 and the additive homomorphism of Pedersen commitments, the individual verification steps pass. Thus J decides against the voter. □

□

Theorem 6. *The protocol has public VVPR verifiability (Definition 10) in the presence of adversary* $\mathcal{A}_{\mathsf{recoverability}}$.

Proof. From the VVPRs (see Figure 5), (rid, v) can be readily derived and it can be matched directly with a corresponding row on $\mathcal{BB}2$. Neither rid nor v can be linked with the voter's take-home receipts. Further, the polling booth information in the VVPR is hidden under a commitment. □

Theorem 7. *The protocol is strongly software independent (Definition 9) in the presence of adversary* $\mathcal{A}_{\mathsf{recoverability}}$.

Proof. We show that for each possible verification failure, we can localise the problem correctly to the failing items and hold the defaulters accountable.

For ballot audit failures, the EA is clearly accountable and the problem is localised to the affected polling booth. For individual verification failures, we can fix accountablity by Theorem 5. For universal verification failures and VVPR audit failures, the error diagnosis and recovery protocols are exhaustively listed in Table II. □

VIII. Practicalities of Implementation

We implemented the core components of the protocol to evaluate its feasibility in real elections. We chose the BN254 elliptic curve [38] to instantiate the pairing groups (G_1, G_2). The BN254 curve is defined over a finite field of 254 bits and provides a basic level of security of 100 bits [39]. (For a production system, a curve with at least 128 bits security should be used.) We used the Charm cryptographic library [40], with a PBC library [41] backend, to code the protocol.

Table III shows various space/time benchmarks for a dummy election with $n = 10^6$ voters and $m = 20$ candidates.

We first note that all ballot and proof sizes are reasonable: modern QR codes can fit around 3 KB of data and other items require moderate bandwidth. We exploit the embarrassingly parallel nature of our ZKP proofs to speed-up computations at the EA and the verifier performing universal verification. Individual verification, which only requires set membership proofs for a single commitment, requires a couple of minutes for the verifier (including signature generation) and a few seconds for the EA.

For future work, we remark that the set membership proofs may also be implemented using generic NIZK proof-systems such as Bulletproofs [43] using suitable circuit encodings of the proof-of-knowledge statements. This would avoid the need to generate different proofs for each verifier while retaining the per-row recoverability property of our proofs.

TABLE III
Benchmarks for 20 candidates and 10^6 voters. [EA]: EA tasks, [V]: verifier tasks. EA always runs in parallel on a compute cluster of 24 nodes, each node with 10 cores. Verifier runs serially for individual verification and on a compute cluster of the same size as EA for universal verification.

Max. size of ballot/receipt QR codes	1.34 KB
Size of $\mathcal{BB}1$	91.5 MB
Size of $\mathcal{BB}2$	29.5 MB
ZKP of set membership, $\lvert\Phi\rvert = 10^6$ (Sec. IV-B):	
- [V] Generating all verifier signatures (serial)	137 s
- [EA] Verifying all verifier signatures[1] (parallel)	2.9 s
- Total size of all verifier signatures	30.5 MB
- [EA] Generating proof for 1 commitment	0.026 s
- [V] Verifying proof for 1 commitment	0.045 s
- Size of NIZK proof for 1 commitment	1248 bits
- [EA] Generating proofs for all the commitments (parallel)	190 s
- [V] Verifying proofs for all the commitments (parallel)	480 s
- Total size of proofs for all the commitments	149 MB
ZKP of reverse set membership, $\lvert\Phi'\rvert = 10^6$ (Sec. IV-C):	
- [EA] Generating all PoKs of commitments[2] (parallel)	2.8 s
- [V] Verifying all PoKs of commitments (parallel)	490 s
- Total size of all PoKs of commitments	88.7 MB
- [V] Generating all verifier signatures (parallel)	470 s
- [EA] Verifying all verifier signatures[1] (parallel)	12 s
- Total size of all verifier signatures	89.6 MB
- [EA] Generating proofs for all the plaintexts (parallel)	410 s
- [V] Verifying proofs for all the plaintexts (parallel)	400 s
- Total size of proofs for all the plaintexts	268 MB

[1]Optimised using the batch verification techniques suggested in [42].
[2]One-time, for all verifiers.

IX. Conclusions

We have presented a DRE-based voting protocol that is not only E2E-V but also recoverable from verification failures. In particular, the protocol supports paper audit trails in a closely coupled and verifiable manner, and is strongly software independent. The DRE frontend avoids voter-initiated challenges and thus targeted attacks against marginalised voters. Our protocol is coercion-resistant. We have also presented a strategy to strengthen vote secrecy from the DRE machine and malicious insiders at the election authority.

The protocol's verifiability, secrecy and recoverability properties have been formally established. The protocol is scalable, efficient and easy to implement.

References

[1] NDI, "The Constitutionality of Electronic Voting in Germany," https://www.ndi.org/e-voting-guide/examples/constitutionality-of-electronic-voting-germany, 2019, [Accessed June 8, 2019].

[2] P. B. Stark, "Conservative statistical post-election audits," in *Nov 15 2007. http://statistics.berkeley.edu/˜stark/Preprints/conservativeElectionAudits07.pdf*, 2007.

[3] P. B. Stark, "Super-Simple Simultaneous Single-Ballot Risk-Limiting Audits," in *2010 Electronic Voting Technology Workshop/ Workshop on Trustworthy Elections (EVT/WOTE 10)*. Washington, DC: USENIX Association, Aug. 2010. [Online]. Available: https://www.usenix.org/conference/evtwote-10/super-simple-simultaneous-single-ballot-risk-limiting-audits

[4] J. Benaloh, D. Jones, E. L. Lazarus, M. Lindeman, and P. B. Stark, "SOBA: Secrecy-preserving Observable Ballot-level Audit," in *2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 11)*. San Francisco, CA: USENIX Association, Aug. 2011. [Online]. Available: https://www.usenix.org/conference/evtwote-11/soba-secrecy-preserving-observable-ballot-level-audit

[5] M. Lindeman, P. B. Stark, and V. S. Yates, "BRAVO: Ballot-polling Risk-limiting Audits to Verify Outcomes," in *2012 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 12)*. Bellevue, WA: USENIX Association, Aug. 2012. [Online]. Available: https://www.usenix.org/conference/evtwote12/workshop-program/presentation/lindeman

[6] P. B. Stark and D. A. Wagner, "Evidence-based elections," *IEEE Secur. Priv.*, vol. 10, no. 5, pp. 33–41, 2012. [Online]. Available: https://doi.org/10.1109/MSP.2012.62

[7] M. Bernhard, J. Benaloh, J. A. Halderman, R. L. Rivest, P. Y. A. Ryan, P. B. Stark, V. Teague, P. L. Vora, and D. S. Wallach, "Public evidence from secret ballots," in *Electronic Voting - Second International Joint Conference, E-Vote-ID 2017, Bregenz, Austria, October 24-27, 2017, Proceedings*, 2017, pp. 84–109. [Online]. Available: https://doi.org/10.1007/978-3-319-68687-5_6

[8] J. Benaloh, "Administrative and public verifiability: Can we have both?" in *Proceedings of the Conference on Electronic Voting Technology*, ser. EVT'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 5:1–5:10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1496739.1496744

[9] S. Bell, J. Benaloh, M. D. Byrne, D. Debeauvoir, B. Eakin, P. Kortum, N. McBurnett, O. Pereira, P. B. Stark, D. S. Wallach, G. Fisher, J. Montoya, M. Parker, and M. Winn, "Star-vote: A secure, transparent, auditable, and reliable voting system," in *2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13)*. Washington, D.C.: USENIX Association, 2013. [Online]. Available: https://www.usenix.org/conference/evtwote13/workshop-program/presentation/bell

[10] N. Farhi, "An Implementation of Dual (Paper and Cryptograhic) Voting System," Master's thesis, The Blatavnik School of Computer Sciences, Tel Aviv University, Tel Aviv University, 2013, https://www.cs.tau.ac.il/~amnon/Students/niko.farhi.pdf.

[11] J. Benaloh, "Ballot casting assurance via voter-initiated poll station auditing," in *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, ser. EVT'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 14–14. [Online]. Available: http://dl.acm.org/citation.cfm?id=1323111.1323125

[12] D. Chaum, "Secret-ballot receipts: True voter-verifiable elections," *IEEE Security and Privacy*, vol. 2, no. 1, pp. 38–47, Jan. 2004. [Online]. Available: http://dx.doi.org/10.1109/MSECP.2004.1264852

[13] J. Heather and D. Lundin, "The append-only web bulletin board," in *Formal Aspects in Security and Trust*, P. Degano, J. Guttman, and F. Martinelli, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 242–256.

[14] R. L. Rivest, "On the notion of software independence in voting systems," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3759–3767, 2008. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2008.0149

[15] R. T. Mercuri, "Physical verifiability of computer systems," in *In International Computer Virus and Security Conference*, 1992. [Online]. Available: http://www.notablesoftware.com/PENN2008/PhysVerify.pdf

[16] B. Adida and R. L. Rivest, "Scratch & vote: Self-contained paper-based cryptographic voting," in *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*, ser. WPES '06. New York, NY, USA: ACM, 2006, pp. 29–40. [Online]. Available: http://doi.acm.org/10.1145/1179601.1179607

[17] F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. R. Savagaonkar, "Innovative Instructions and Software Model for Isolated Execution," in *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*, ser. HASP '13. New York, NY, USA: Association for Computing Machinery, 2013. [Online]. Available: https://doi.org/10.1145/2487726.2488368

[18] A. Juels, D. Catalano, and M. Jakobsson, "Towards trustworthy elections," D. Chaum, M. Jakobsson, R. L. Rivest, P. A. Ryan, and J. Benaloh, Eds. Berlin, Heidelberg: Springer-Verlag, 2010, ch. Coercion-resistant Electronic Elections, pp. 37–63. [Online]. Available: http://dl.acm.org/citation.cfm?id=2167913.2167915

[19] A. Essex and J. Clark, "Punchscan in practice: an e2e election case study," in *Proceedings of IAVoSS Workshop on Trustworthy Elections (WOTE)*, Ottowa, Canada, 2007.

[20] P. Y. A. Ryan, D. Bismark, J. Heather, S. Schneider, and Z. Xia, "Prêt à voter: A voter-verifiable voting system," *Trans. Info. For. Sec.*, vol. 4, no. 4, pp. 662–673, Dec. 2009. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2009.2033233

[21] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. Sherman, and P. Vora, "Scantegrity: End-to-end voter-verifiable optical-scan voting," *IEEE Security and Privacy*, vol. 6, no. 3, pp. 40–46, May 2008.

[22] D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. L. Rivest, P. Y. A. Ryan, E. Shen, and A. T. Sherman, "Scantegrity II: End-to-End Verifiability for Optical Scan Election Systems Using Invisible Ink Confirmation Codes," in *Proceedings of the Conference on Electronic Voting Technology*, ser. EVT'08. USA: USENIX Association, 2008.

[23] A. T. Sherman, R. A. Fink, R. Carback, and D. Chaum, "Scantegrity III: automatic trustworthy receipts, highlighting over/under votes, and full voter verifiability," in *2011 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '11, San Francisco, CA, USA, August 8-9, 2011*, H. Shacham and V. Teague, Eds. USENIX Association, 2011. [Online]. Available: https://www.usenix.org/conference/evtwote-11/scantegrity-iii-automatic-trustworthy-receipts-highlighting-overunder-votes

[24] B. Adida and C. A. Neff, "Efficient receipt-free ballot casting resistant to covert channels," in *Proceedings of the 2009 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, ser. EVT/WOTE'09. Berkeley, CA, USA: USENIX Association, 2009, pp. 11–11. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855491.1855502

[25] J.-M. Bohli, J. Müller-Quade, and S. Röhrich, "Bingo voting: Secure and coercion-free voting using a trusted random number generator," in *Proceedings of the 1st International Conference on E-voting and Identity*, ser. VOTE-ID'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 111–124. [Online]. Available: http://dl.acm.org/citation.cfm?id=1787456.1787470

[26] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems," in *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, ser. STOC '85. New York, NY, USA: ACM, 1985, pp. 291–304. [Online]. Available: http://doi.acm.org/10.1145/22145.22178

[27] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981. [Online]. Available: http://doi.acm.org/10.1145/358549.358563

[28] M. Jakobsson, A. Juels, and R. L. Rivest, "Making mix nets robust for electronic voting by randomized partial checking," in *Proceedings of the 11th USENIX Security Symposium*. USA: USENIX Association, 2002, p. 339–353.

[29] T. Kaczmarek, J. Wittrock, R. Carback, A. Florescu, J. Rubio, N. Runyan, P. L. Vora, and F. Zagórski, "Dispute resolution in accessible voting systems: The design and use of audiotegrity," in *E-Voting and Identify - 4th International Conference, VoteID 2013, Guildford, UK, July 17-19, 2013. Proceedings*, ser. Lecture Notes in Computer Science, J. Heather, S. A. Schneider, and V. Teague, Eds., vol. 7985. Springer, 2013, pp. 127–141. [Online]. Available: https://doi.org/10.1007/978-3-642-39185-9_8

[30] S. Popoveniuc and A. Regenscheid, "Sigma ballots," in *Electronic Voting 2010, EVOTE 2010, 4th International Conference, Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting.CC, July 21st - 24th, 2010, in Castle Hofen, Bregenz, Austria*, ser. LNI, R. Krimmer and R. Grimm, Eds., vol. P-167. GI, 2010, pp. 179–190. [Online]. Available: https://dl.gi.de/20.500.12116/19492

[31] D. Lundin and P. Y. A. Ryan, "Human Readable Paper Verification of Prêt à Voter," in *Computer Security - ESORICS 2008*, S. Jajodia and J. Lopez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 379–395.

[32] A. Essex, C. Henrich, and U. Hengartner, "Single Layer Optical-Scan Voting with Fully Distributed Trust," in *E-Voting and Identity*, A. Kiayias and H. Lipmaa, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 122–139.

[33] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology - EUROCRYPT 2004*, C. Cachin and J. L.

Camenisch, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 56–73.
[34] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '91. London, UK, UK: Springer-Verlag, 1992, pp. 129–140. [Online]. Available: http://dl.acm.org/citation.cfm?id=646756.705507
[35] J. Camenisch, R. Chaabouni, and A. Shelat, "Efficient protocols for set membership and range proofs," in *Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, ser. ASIACRYPT '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 234–252. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-89255-7_15
[36] M. H. Au, W. Susilo, and Y. Mu, "Constant-Size Dynamic k-TAA," in *Security and Cryptography for Networks*, R. De Prisco and M. Yung, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 111–125.
[37] J. Clark and U. Hengartner, "On the use of financial data as a random beacon," in *Proceedings of the 2010 International Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, ser. EVT/WOTE'10. USA: USENIX Association, 2010, p. 1–8.
[38] P. S. L. M. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected Areas in Cryptography*, B. Preneel and S. Tavares, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 319–331.
[39] S. Yonezawa, T. Kobayashi, and T. Saito, "Pairing-Friendly Curves," https://tools.ietf.org/id/draft-yonezawa-pairing-friendly-curves-02.html, 2019, [(work in progress); accessed July 28, 2021].
[40] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013. [Online]. Available: http://dx.doi.org/10.1007/s13389-013-0057-3
[41] B. Lynn, "PBC Library (pbc-0.5.14)," https://crypto.stanford.edu/pbc/, 2013, [Accessed June 10, 2019].
[42] A. L. Ferrara, M. Green, S. Hohenberger, and M. Ø. Pedersen, "Practical short signature batch verification," in *Topics in Cryptology – CT-RSA 2009*, M. Fischlin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 309–324.
[43] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," Cryptology ePrint Archive, Report 2017/1066, 2017, https://ia.cr/2017/1066.
[44] A. Fiat and A. Shamir, "How To Prove Yourself: Practical Solutions to Identification and Signature Problems," in *Advances in Cryptology — CRYPTO' 86*, A. M. Odlyzko, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 186–194.
[45] M. H. Au, W. Susilo, and Y. Mu, "Constant-Size Dynamic k-TAA," in *Security and Cryptography for Networks*, R. De Prisco and M. Yung, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 111–125.

APPENDIX A
ZKP OF SET MEMBERSHIP: C IS A COMMITMENT OF SOME $\rho \in \Phi$

The treatment follows [35]. We provide a variant for the asymmetric pairing case in Figure 7.

The ZKP protocol relies on the Boneh-Boyen short signature scheme [33]. The secret key of the signer is $x \leftarrow \mathbb{Z}_q$, the corresponding public key is $y = g_2^x$. The signature on a message ρ is $\sigma \leftarrow g_1^{1/(x+\rho)}$; verification is done by checking that $e(\sigma, y \cdot g_2^{\rho}) = e(g_1, g_2)$.

Theorem 8 (adapted from [35]). *If the $|\Phi|$-Strong Diffie-Hellman assumption holds in (G_1, G_2), then the protocol in Figure 7 is an honest-verifier zero-knowledge argument that commitment C commits a member of set Φ, i.e., the prover knows ρ, r such that $C = g_1^{\rho} h_1^{r}$ and $\rho \in \Phi$.*

To convert honest-verifier zero-knowledge to full zero-knowledge, the prover additionally needs to verify all verifier signatures. Additionally, the proof of knowledge can be made general zero-knowledge in the random oracle model by employing the Fiat-Shamir heuristic [44], where the verifier's challenge is replaced by the output of the hash function on the input and the prover's first message.

APPENDIX B
ZKP OF REVERSE SET MEMBERSHIP: ρ IS COMMITTED BY SOME C IN Φ'

For this proof, the prover obtains verifier's signatures on committed values by presenting to her only the commitments, along with a proof of knowledge of the committed values. Later, a proof of knowledge of a signature on the committed message proves that the message ρ was committed by one of the presented commitments.

BBS+ signatures are particularly well suited for this task. We first recall their definition from [45].

Definition 12 (BBS+ signatures (adapted from [45])). *A BBS+ signature scheme is given by algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *specified as follows:*

- Setup*: Obtain* $(q, G_1, G_2, G_T, f_1, g_1, h_1, f_2) \leftarrow \mathcal{G}(1^n)$. *$G_1, G_2, G_T$ are cyclic groups of prime order q such there exists an admissible bilinear map $e : G_1 \times G_2 \rightarrow G_T$. f_1, g_1, h_1 are generators of G_1 and f_2 is a generator of G_2.*
- KeyGen*: Randomly choose secret key $x \in \mathbb{Z}_q^*$ and set public key as $y := f_2^x$.*
- $\mathsf{Sign}(m, x)$*: Choose random $c, r \in \mathbb{Z}_q$ and compute $A = (f_1 g_1^m h_1^r)^{\frac{1}{c+x}}$. Output $\sigma := (A, c, r)$.*
- $\mathsf{Verify}(\sigma, m, y)$*: Parse σ as $(A, c, r) := \sigma$. Output 1 if and only if $e(A, y f_2^c) = e(f_1 g_1^m h_1^r, f_2)$.*

Theorem 9 (adapted from [45]). *BBS+ signature is unforgeable against adaptively chosen message attack under the n-SDH assumption, where n represents the number of signature queries made by the adversary.*

Theorem 10. *If the $|\Phi|'$-Strong Diffie-Hellman assumption holds in (G_1, G_2), the protocol in Figure 8 is an honest-verifier zero-knowledge argument (in the random oracle model) that message ρ is committed by one of the commitments in set Φ', i.e., the prover knows an r such that $g_1^{\rho} h_1^{r} \in \Phi'$.*

As with the previous set membership proof, conversion to full zero-knowledge requires the prover to additionally verify all verifier signatures.

Proof. We prove the completeness, soundness and zero-knowledge properties below.

Completeness: For completeness, we must show that if the prover knows an r_i such that $C_i = g_1^{\rho} h_1^{r_i} \in \Phi'$, then it can make an honest verifier accept by following the protocol honestly. We show that in this case all the conditions of the proof of knowledge π_{sig} are satisfied and hence the verifier accepts because of the completeness of π_{sig}.

Common Input: Groups $G_1 = \langle g_1 \rangle = \langle h_1 \rangle$, $G_2 = \langle g_2 \rangle$ of prime order q, a commitment $C \in G_1$, and set Φ
Prover Input: $\rho, r \in \mathbb{Z}_q$ such that $C = g_1^{\rho} h_1^{r}$ and $\rho \in \Phi$

Prover obtains verifier's signatures on messages $\{\rho_i \in \Phi\}$:

$P \xleftarrow{y, \{\sigma_i\}} V$ Verifier picks random $x \in \mathbb{Z}_q$ and sends $y \leftarrow g_2^x$ and $\sigma_i \leftarrow g_1^{\frac{1}{x+\rho_i}}$ for every $\rho_i \in \Phi$.

Prover sends blinded signature V on message ρ and proves PK$\{(\rho, r, v) : C = g_1^{\rho} h_1^{r} \wedge V = g_1^{\frac{v}{x+\rho}}\}$:

$P \xrightarrow{V} V$ Prover picks random $v \in \mathbb{Z}_p$ and sends $V \leftarrow \sigma_i^v$, where i is the index of message ρ in Φ.
$P \xrightarrow{a, D} V$ Prover picks random $s, t, m \in \mathbb{Z}_q$ and sends $a \leftarrow e(V, g_2)^{-s} e(g_1, g_2)^{t}$ and $D \leftarrow g_1^{s} h_1^{m}$.
$P \xleftarrow{c} V$ Verifier sends a random challenge $c \in \mathbb{Z}_q$.
$P \xrightarrow{z_\rho, z_v, z_r} V$ Prover sends $z_\rho = s - \rho c$, $z_v = t - vc$, and $z_r = m - rc$,
Verifier checks that $D \stackrel{?}{=} C^c h_1^{z_r} g_1^{z_\rho}$ and that $a \stackrel{?}{=} e(V, y)^c \cdot e(V, g_2)^{-z_\rho} \cdot e(g_1, g_2)^{z_v}$

Fig. 7. Set membership protocol for proving that a commitment C commits a member of set Φ [35]

Common Input: Groups $G_1 = \langle f_1 \rangle = \langle g_1 \rangle = \langle h_1 \rangle$, $G_2 = \langle f_2 \rangle$ of prime order q, a message $\rho \in \mathbb{Z}_q$, and a set Φ' of commitments
Prover Input: C, r such that $C = g_1^{\rho} h_1^{r}$ and $C \in \Phi'$

Prover proves knowledge of messages committed by $\{C_i \in \Phi'\}$:

$\mathcal{P} \xrightarrow{\{\pi_{\mathsf{com}_i}\}} \mathcal{V}$ Prover sends the following non-interactive PoK for each $i \in [|\Phi'|]$ (using the Fiat-Shamir heuristic [44]):
$\pi_{\mathsf{com}_i} := PK\{(\rho_i, r_i) : C_i = g_1^{\rho_i} h_1^{r_i}\}$

Prover obtains verifier's signatures on messages committed by $\{C_i \in \Phi'\}$:

$\mathcal{P} \xleftarrow{y, \{\sigma'_i\}} \mathcal{V}$ Verifier:
- generates fresh secret key $x \in \mathbb{Z}_q^*$ and sends public key $y := f_2^x$ to the prover;
- for each $i \in [|\Phi'|]$, generates random c_i, r'_i, computes $A_i := (f_1 h_1^{r'_i} C_i)^{\frac{1}{c_i + x}}$;
- for each $i \in [|\Phi'|]$, sends $\sigma'_i := (A_i, c_i, r'_i)$ to the prover.

Prover (for each $i \in [|\Phi'|]$):
- computes $r''_i = r'_i + r_i$, where r_i denotes the randomness associated with C_i;
- stores $\sigma''_i := (A_i, c_i, r''_i)$ as a BBS+ signature [45] on message ρ_i committed by C_i.

Prover proves knowledge of a verifier's signature on message ρ:

$\mathcal{P} \xrightarrow{B_1, B_2, \pi_{\mathsf{sig}}} \mathcal{V}$ Prover:
- finds index i such that C_i commits the given ρ;
- chooses random $s_1, s_2 \in \mathbb{Z}_q^*$;
- computes $B_1 := g_1^{s_1} h_1^{s_2}$, $B_2 := A_i g_1^{s_2}$, $\delta_1 := s_1 c_i$, $\delta_2 := s_2 c_i$;
- creates the following non-interactive PoK (using $c = c_i$ and $r = r''_i$):
$\pi_{\mathsf{sig}} := PK\{(s_1, s_2, c, r, \delta_1, \delta_2) : B_1 = g_1^{s_1} h_1^{s_2} \wedge B_1^c = g_1^{\delta_1} h_1^{\delta_2} \wedge \frac{e(B_2, y)}{e(f_1, f_2)} = e(B_2, f_2)^{-c} e(g_1, y)^{s_2} e(g_1, f_2)^{\delta_2} e(g_1, f_2)^{\rho} e(h_1, f_2)^{r}\}$
- sends $(B_1, B_2, \pi_{\mathsf{sig}})$ to the verifier.

Verifier accepts if verification of π_{sig} against (B_1, B_2) passes.

Fig. 8. Set membership protocol for proving that ρ is committed by one of the commitments in Φ' (protocols for obtaining BBS+ signatures on committed messages and proving knowledge of signatures adapted from [45]).

Signature $\sigma'_i := (A_i, c_i, r'_i)$ from the verifier must satisfy $A_i = (f_1 h_1^{r'_i} g_1^{\rho} h_1^{r_i})^{\frac{1}{c_i+x}}$. Thus, $A_i^{c_i+x} = f_1 g_1^{\rho} h_1^{r''_i}$. Also, the prover sets $B_2 = A_i g_1^{s_2}$ and $\delta_2 = s_2 c_i$. Thus:

$$
\begin{aligned}
& e(A_i^{c_i+x}, f_2) = e(f_1 g_1^{\rho} h_1^{r''_i}, f_2) \\
\implies & e((B_2 g_1^{-s_2})^{c_i+x}, f_2) = e(f_1, f_2) e(g_1^{\rho}, f_2) e(h_1^{r''_i}, f_2) \\
\implies & e(B_2^{c_i}, f_2) e(B_2^{x}, f_2) e(g_1^{-s_2 c_i}, f_2) e(g_1^{-s_2 x}, f_2) = \\
& \qquad e(f_1, f_2) e(g_1, f_2)^{\rho} e(h_1, f_2)^{r''_i} \\
\implies & \frac{e(B_2^{x}, f_2)}{e(f_1, f_2)} = \\
& e(B_2^{-c_i}, f_2) e(g_1^{s_2 x}, f_2) e(g_1^{s_2 c_i}, f_2) e(g_1, f_2)^{\rho} e(h_1, f_2)^{r''_i} \\
\implies & \frac{e(B_2, y)}{e(f_1, f_2)} = \\
& e(B_2, f_2)^{-c_i} e(g_1, y)^{s_2} e(g_1, f_2)^{\delta_2} e(g_1, f_2)^{\rho} e(h_1, f_2)^{r''_i}
\end{aligned}
\tag{2}
$$

Next, setting $B_1 := g_1^{s_1} h_1^{s_2}$, $\delta_1 := s_1 c_i$ and $\delta_2 := s_2 c_i$ implies $B_1^{c_i} = g_1^{s_1 c_i} h_1^{s_2 c_i} = g^{\delta_1} h^{\delta_2}$. Thus, all the equations of π_{sig} are satisfied as desired.

Soundness: For soundness, we show that if the verifier accepts then an extractor can extract an r such that $g_1^{\rho} h_1^{r} \in \Phi'$. Our extractor runs the extractor for π_{com_i} to obtain (ρ_i, r_i) for each $i \in [|\Phi'|]$.

Case 1 ($\exists i^*, \rho_i^* = \rho$): In this case, the extractor simply outputs r_{i^*} as the desired value.

Case 2 ($\forall i, \rho_i \neq \rho$): We show that this case is not possible because of computational binding of Pedersen commitments and unforgeability of BBS+ signatures under the $|\Phi'|$-Strong Diffie-Hellman assumption.

We construct a forger $\mathcal{F}$ interacting with a BBS+ signing oracle $\mathcal{S}$. The forger internally interacts with a malicious prover $\mathcal{P}^*$. $\mathcal{F}$ and $\mathcal{P}^*$ additionally obtain the common inputs of the set membership protocol. The unforgeability game between $\mathcal{F}$ and $\mathcal{S}$ goes as follows:

1) $\mathcal{S}$ runs the Setup algorithm to obtain generators f_1, g_1, h_1, f_2 and other public parameters. It generates random secret key $x \in \mathbb{Z}_q$ and sends public key $y = f_2^x$ and the public parameters to $\mathcal{F}$.
2) $\mathcal{F}$ obtains π_{com_i} from $\mathcal{P}^*$ and runs for each $i \in [|\Phi'|]$ the extractor for π_{com_i} to extract (ρ_i, r_i).
3) $\mathcal{F}$ sends signature queries ρ_i to $\mathcal{S}$ and obtains signatures $\sigma'_i := (A_i, c_i, r'_i)$. It then sends $y, \{\sigma'_i\}$ to $\mathcal{P}^*$.
4) On obtaining $(B_1, B_2, \pi_{\mathsf{sig}})$ from $\mathcal{P}^*$, $\mathcal{F}$ runs the extractor for π_{sig} and obtains $(s_1, s_2, c, r, \delta_1, \delta_2)$.
5) $\mathcal{F}$ computes $m^* = \rho$, $\sigma^* = (B_2 g_1^{-s_2}, c, r)$ and outputs (m^*, σ^*) as its forgery.

By the soundness of π_{sig}, $(s_1, s_2, c, r, \delta_1, \delta_2)$ are such that:

$$B_1 = g_1^{s_1} h_1^{s_2} \tag{3}$$

$$B_1^{c} = g_1^{\delta_1} h_1^{\delta_2} \tag{4}$$

$$
\begin{aligned}
& \frac{e(B_2, y)}{e(f_1, f_2)} = \\
& e(B_2, f_2)^{-c} e(g_1, y)^{s_2} e(g_1, f_2)^{\delta_2} e(g_1, f_2)^{\rho} e(h_1, f_2)^{r}
\end{aligned}
\tag{5}
$$

Equations 3 and 4, together with the computational binding property of Pedersen commitments, imply the following:

$$\delta_1 = s_1 c \qquad \delta_2 = s_2 c \tag{6}$$

Substituting $\delta_2 = s_2 c$ in Equation 5 (and proceeding in the direction opposite to the derivation of Equation 2), we get:

$$
\begin{aligned}
& \frac{e(B_2, f_2^{x})}{e(f_1, f_2)} = \\
& e(B_2, f_2)^{-c} e(g_1, f_2^{x})^{s_2} e(g_1, f_2)^{s_2 c} e(g_1, f_2)^{\rho} e(h_1, f_2)^{r} \\
\implies & e((B_2 g_1^{-s_2})^{c+x}, f_2) = e(f_1 g_1^{\rho} h_1^{r}, f_2) \\
\implies & B_2 g_1^{-s_2} = (f_1 g_1^{\rho} h_1^{r})^{\frac{1}{c+x}}
\end{aligned}
\tag{7}
$$

Thus, $\sigma^* = (B_2 g_1^{-s_2}, c, r)$ is a valid signature on message $m^* = \rho$ under public key $y = f_2^x$ because it satisfies the BBS+ signature verification equation:

$$
\begin{aligned}
e(B_2 g_1^{-s_2}, y f_2^{c}) &= e((f_1 g_1^{\rho} h_1^{r})^{\frac{1}{c+x}}, f_2^{x} f_2^{c}) \\
&= e(f_1 g_1^{\rho} h_1^{r}, f_2)^{\frac{c+x}{c+x}} = e(f_1 g_1^{\rho} h_1^{r}, f_2)
\end{aligned}
\tag{8}
$$

Since $\forall i, \rho_i \neq \rho$, no signature on message ρ was queried from $\mathcal{S}$ in step 3 and this is a forgery. Since the forger queries $|\Phi'|$ signatures in total, this is not possible under the $|\Phi'|$-Strong Diffie-Hellman assumption (Theorem 9).

Honest-verifier zero-knowledge: To show honest-verifier zero-knowledge, we construct a simulator that produces a transcript for the verifier indistinguishable from the real transcript. Our simulator works as follows:

1) Invoke simulators for $\{\pi_{\mathsf{com}_i}\}$ to obtain simulated proofs $\{\pi^S_{\mathsf{com}_i}\}$. Send $\{\pi^S_{\mathsf{com}_i}\}$ to the verifier.
2) Choose random $s_1, s_2, s_3 \in \mathbb{Z}_q^*$ and set $B_1^S := g_1^{s_1} h_1^{s_2}$ and $B_2^S := g_1^{s_3}$. Invoke the simulator for π_{sig} to obtain simulated proof π^S_{sig}. Send $(B_1^S, B_2^S, \pi^S_{\mathsf{sig}})$ to the verifier.

The simulated proofs are indistinguishable from the real ones because of honest-verifier zero-knowledge property of $\{\pi_{\mathsf{com}_i}\}$ and π_{sig}. B_1^S is indistinguishable from B_1 because of perfect hiding of Pedersen commitments. Since s_2 is perfectly hidden by B_1, B_2^S is indistinguishable from B_2. Thus the entire protocol is honest-verifier zero-knowledge. □