DeVoS: Deniable Yet Verifiable Vote Updating

Johannes Müller
University of Luxembourg
johannes.mueller@uni.lu

Balázs Pejó
Budapest University of Technology and Economics
pejo@crysys.hu

Ivan Pryvalov
University of Luxembourg & Brandenburg University of Technology
ivan.pryvalov@b-tu.de

ABSTRACT

Internet voting systems are supposed to meet the same high standards as traditional paper-based systems when used in real political elections: freedom of choice, universal and equal suffrage, secrecy of the ballot, and independent verifiability of the election result. Although numerous Internet voting systems have been proposed to achieve these challenging goals simultaneously, few come close in reality.

We propose a novel publicly verifiable and practically efficient Internet voting system, DeVoS, that advances the state of the art. The main feature of DeVoS is its ability to protect voters' freedom of choice in several dimensions. First, voters in DeVoS can intuitively update their votes in a way that is deniable to observers but verifiable by the voters; in this way voters can secretly overwrite potentially coerced votes. Second, in addition to (basic) vote privacy, DeVoS also guarantees strong participation privacy by end-to-end hiding which voters have submitted ballots and which have not. Finally, DeVoS is fully compatible with Perfectly Private Audit Trail, a state-of-the-art Internet voting protocol with practical everlasting privacy. In combination, DeVoS offers a new way to secure free Internet elections with strong and long-term privacy properties.

KEYWORDS

electronic voting, everlasting privacy, verifiability, receipt-freeness, participation privacy

1 INTRODUCTION

Determining the will of the people is the essence of free elections. However, this freedom must be protected from various threats, in particular the following. Electoral authorities can be influenced by the government or other powers to covertly destroy legitimate ballots or stuff illegitimate ones. People, especially members of marginalized groups, can be intimidated to discourage them from voting. Wealthy actors can buy people's votes to win their favor. Because such threats are real, the *United Nations (UN)* [45] *International Human Rights Standards on Elections* require that political elections be independently *verifiable* (§127), that voters' ballots be *secret* (§16), and that voters be *protected from any form of coercion or compulsion* to disclose how they intend to vote or how they have voted (§92).

These basic requirements for free elections apply to all types of voting systems, including those that allow voters to submit digital ballots over the Internet. Such systems have been used for real political elections in Australia, Estonia, France, Norway, and Switzerland, to name but a few.

State of practice. To our knowledge, however, there are only two Internet voting systems used for political elections that claim to meet at least some of these basic requirements. But even these two systems, the ones used in Estonia and Switzerland, do not meet these requirements simultaneously or to a sufficient degree, respectively, as we explain next.

The Estonian Internet voting system IVXV [32] was designed to guarantee the privacy of the vote, and in order to avoid possible coercion, IVXV offers voters the possibility to update their previously submitted ballots in a deniable way. However, the digital ballot trail in IVXV is only partially verifiable, and even some of the supposedly verifiable parts have been shown not to be [46]. Furthermore, IVXV does not provide vote privacy under the assumptions originally stated by its developers [43].

The Internet voting system used for political elections in Switzerland has been revised after several serious security problems were discovered [27]. Various auditors from academia and industry have analyzed the revised version of the Swiss Internet voting system to basically confirm that it provides the intended security features [22]: public verifiability and privacy of votes. However, this system was not designed to protect against malicious actors who want to influence elections by intimidating voters or buying their votes.

Although protection against any form of coercion or compulsion, as demanded by the UN (see above), appears to be practically unattainable for the entire electorate, both for Internet and paper-based voting systems, this state of practice is disappointing. Malicious voter influence should be made as inefficient as possible to reduce its impact on the final election outcome without compromising verifiability. Fortunately, as we recall below, the state of research on this challenging problem is better than the state of the practice, albeit with significant room for improvement.

State of research. For more than two decades, researchers have been searching for technical solutions to prevent voters from being influenced in their free formation of opinion when casting their votes in secure Internet voting systems. It has proved extremely difficult, if not impossible, to find a patent solution. In fact, the academic literature on the subject is very extensive and contains many different proposals, based on different assumptions, with different approaches and different objectives.

Some of these proposals prevent voters' local data, generated during electronic ballot casting, from inadvertently serving as evidence of their vote. This property is called *receipt-freeness*, and relevant work in this area includes, among others, [8, 36]. Other work, such as Selene [48], addresses the problem of how voters can use personal codes to verify that the vote they cast is indeed

Proceedings on Privacy Enhancing Technologies 2024(1), 357–378

https://doi.org/10.56553/popets-2024-0021

included in the final result, without these codes being able to serve as proof of the voter's vote to third parties. What these papers have in common is the assumption that voters are honest and hence do not deviate from their prescribed program to produce evidence that can serve as convincing proof to a coercer or vote buyer.

In this paper, however, we are interested in exactly this problem: how to prevent a voter from being able to convince anyone of her voting behavior, even if she actively tries to do so, e.g. because she is coerced or wants to sell her vote? Many papers have addressed this challenge. For example, in BeleniosRF [5], even those voters who try to sell their vote by using random coins chosen by a potential vote buyer do not obtain any cryptographic proof of their choices. In protocols such as JCJ [34]/Civitas [9], coerced voters can choose fake credentials that the coercer cannot distinguish from the real ones, but which ensure that the coerced votes are secretly removed by the tellers. In other schemes, such as VoteAgain [31, 41], voters can overwrite their potentially coerced votes in a way that is deniable to the coercer but verifiable by the voters. We will discuss the features of these and other works in more detail in Section 2.

Beyond the individual pros and cons of the various proposals, a recent systematic review has identified a global problem [28]: there is no secure and efficient Internet voting system that can simultaneously mitigate the effects of malicious influence and keep votes secret in the long term. In fact, in all state-of-the-art systems that limit coercion or vote buying, such as BeleniosRF, JCJ/Civitas or VoteAgain, votes are encrypted using the tellers' public key and the resulting ciphertexts are then posted on a bulletin board for verification purposes. Since there can be no unconditional secrecy in a public-key setting, these votes are secret only under certain hardness assumptions (e.g. Decisional Diffie Hellman in the case of ElGamal PKE). However, since any observer can read these encrypted votes and store them for any length of time, secrecy could be undermined retrospectively by new cryptanalytic methods or more powerful computers (e.g. quantum computers).

This state of research is unsatisfactory because many elections require confidentiality to be guaranteed not just for some time, but for several decades because voters may have to fear negative consequences even if their individual choices are revealed, say, 10 or 20 years after the election. Instead, electronic voting systems should provide *everlasting privacy* [28], i.e. privacy without relying on any hardness assumption. However, it is far from obvious how existing systems (e.g. Civitas, BeleniosRF or VoteAgain) with strong privacy features can be modified to provide everlasting privacy and still be practically efficient. This is because these systems use specific cryptographic primitives (e.g. *re-randomizable signatures* as in BeleniosRF) or their overall protocols are highly complex (as in Civitas or VoteAgain), which makes it difficult to combine them with state-of-the-art Internet voting protocols that provide everlasting privacy, which deploy specific cryptographic primitives themselves (e.g. *commitment-consistent encryption* as in [15]).

Our contributions. We propose DeVoS, a novel publicly verifiable and practically efficient Internet voting protocol that advances the state of the art as described below. In fact, DeVoS provides the following unique combination of privacy properties:

(1) *Deniable vote updating:* DeVoS enables voters to secretly overwrite their potentially coerced votes in an intuitive way.
(2) *Strong participation privacy:* DeVoS hides which voters did and did not vote, even if the voters who did vote try to convince a potential coercer otherwise.
(3) *Practical everlasting privacy:* DeVoS is fully compatible with *Perfectly Private Audit Trail (PPAT)* [15], a state-of-the-art verifiable Internet voting protocol with everlasting privacy from the public (according to [28]). In this way, DeVoS can be extended to make all of its privacy features (i.e., deniable vote updating, strong participation privacy, and basic vote privacy) unconditionally private from the public, and thus long-term.

DeVoS achieves these properties under realistic assumptions about the election infrastructure (in particular, no anonymous submission channels) and under trust assumptions that are equivalent to state-of-the-art verifiable Internet voting systems with similar strong privacy features (e.g., BeleniosRF, Civitas, or VoteAgain). In particular, even if the voters and the tallying authority are fully malicious, the correctness of the result can still be publicly verified.

Since DeVoS follows the same approach as IVXV to mitigate the risk of coercion, but unlike IVXV provides full public verifiability, DeVoS offers a practical option to make political elections in Estonia more secure.

In addition to proposing DeVoS, we have formally analyzed its security in an established cryptographic security framework for e-voting protocols. We have also implemented the key cryptographic components of DeVoS and provide detailed benchmarks to demonstrate its practical efficiency.

Finally, we study the trade-offs between deniability/participation privacy and efficiency of DeVoS. We propose different optimization strategies to increase efficiency while providing a relaxed yet sufficient deniability/participation privacy guarantee. We analyze these trade-offs using techniques from differential privacy.

Overview of paper. In Section 2, we elaborate on related Internet voting protocols. In Section 3, we illustrate DeVoS and its features, we specify the threat scenario and discuss how DeVoS relates to the state of the art. In Section 4 we present DeVoS with full technical details. In Section 5, we state the security and privacy properties of DeVoS. In Section 6, we present two instantiations of the abstract DeVoS protocol, one with conditional and one with unconditional (aka everlasting) privacy from the public. In Section 7, we present efficiency benchmarks for our instantiations of DeVoS. We conclude in Section 8. We provide further technical details in the Appendix.

2 RELATED WORK

In the following, we briefly review the state of the art on secure Internet voting systems that mitigate the effect of malicious influence of the voters.

Fake credentials. One prominent approach is the deployment of *fake credentials*. In such Internet voting systems, voters have two options when generating their digital ballot: (1) if the voter is free from coercion, she uses her correct credential to generate and submit a ballot for her preferred candidate, (2) if the voter is coerced to vote for another candidate, she makes up a fake credential and uses it to generate a ballot for the forced candidate. Since ballots with fake credentials are secretly but verifiably removed during the tallying process, a coercer cannot tell whether or not the voter has ultimately voted for the coerced candidate. In addition, it remains secret which voters participated in the election.

Fake credentials are used in a number of Internet voting protocols, such as JCJ [34] which was later implemented as Civitas [9].

While fake credentials work in theory, their practical value is limited. First, voters have to remember long, random credentials and enter them correctly, and second, the complexity of the tallying phase is such that only elections with small electorates could be run efficiently. There have been some attempts to mitigate these practical limitations (see, for example, [21]), but these usually come at the cost of weakening the security or privacy features.

Re-randomizable signatures. In Internet voting systems with *re-randomizable signatures*, such as BeleniosRF [5], the voters' digital ballots are re-randomized before being posted to the public bulletin board. With this re-randomization, the random coins used by the voter to encrypt their ballot do not provide any evidence that the voter has cast a particular vote. This feature, called *strong receipt-freeness*, protects against voters being able to sell their local data as a convincing receipt to potential vote buyers, even if the voters deviate from their honest program to create such evidence. In addition to the low computational overhead, another advantage of re-randomizable signatures is that they, unlike fake credentials, do not complicate the voters' casting process.

However, this approach has two major limitations. First, it reveals which voters participated in the election, which is not, for example, fully in line with the guidelines of the Venice Commission.[1] Second, this approach does not provide any resistance against the kind of 'primitive' influence where a potential on-site coercer watches a voter enter a vote into the computer and can thus easily verify that the voter is following instructions.

Deniable vote updating. Deniable vote updating is another approach to mitigating the effects of coercion. In such systems, for example VoteAgain [31, 41], voters can overwrite their potentially coerced votes in a way that is deniable to the coercer but verifiable by the voters. This feature is typically realized by inserting the encrypted ballots into a swarm of "dummy" ballots, which are then secretly but verifiably canceled out along with the overwritten ballots before the tallying. Essentially the same mechanism also hides which voters actually participated in the election, even if abstaining voters want to prove to a coercer that they have abstained; we call this feature, *strong participation privacy.*

On a practical level, the strategies for human voters to defeat coercion in systems with deniable vote updating are more intuitive than with fake credentials: voters who are coerced to cast a particular vote can cast a new vote after the coercer has left, and voters who are coerced to abstain from voting can cast a vote whenever the coercer is absent.

The main limitation of deniable vote updating is the assumption that the coercion-free time window, which is generally required to protect against coercion, is more restricted than in the fake credential approach: in fact, the coercion-free time must lie between the presence of the coercer and the end of the submission phase in order for voters to secretly overwrite their votes.

3 OVERVIEW

We describe the main idea of DeVoS and explain at an intuitive level why it guarantees the features we claim. We also specify the threat scenario we consider and how DeVoS relates with the state of the art. Since DeVoS can extend any basic secure e-voting system, we first recall the concept of such systems.

3.1 Basic secure e-voting

DeVoS is compatible with both existing approaches to secure electronic ballot counting: homomorphic aggregation and verifiable shuffling. In the following, we describe the common basic structure of these two approaches, so that we can explain below how DeVoS extends them.

Cryptography. The basic cryptographic component is an IND-CPA-secure public-key encryption scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$. In order to tally the ciphertexts of this scheme, they need to be *malleable* in the following sense.

If verifiable shuffling is used to shuffle ballots, then ciphertexts must be *re-randomizable*, meaning that a ciphertext $e' = \mathsf{Enc}(pk, m; r')$ with 'fresh' randomness r' can be efficiently computed from a ciphertext $e = \mathsf{Enc}(pk, m; r)$ that encrypts the same message m but with different randomness r. This computation does neither require knowledge of the secret key sk, nor of the message m, nor of the randomness r.

When ballots are tallied homomorphically, then the ciphertexts have to be *additively homomorphic*: Given two ciphertexts $e_1 = \mathsf{Enc}(pk, m_1)$ and $e_2 = \mathsf{Enc}(pk, m_2)$, a ciphertext $e_3 = \mathsf{Enc}(\mathsf{pk}, m_3)$ can be computed efficiently that encrypts the sum $m_3 = m_1 + m_2$, without knowledge of the secret key sk or the messages m_1, m_2.

The most common implementation of this primitive is ElGamal PKE and its variations.

Various *non-interactive zero-knowledge proofs (NIZKPs)* are used to allow parties to produce convincing evidence that can be verified by anyone to check that these parties have processed their data correctly (this is called *soundness*), without revealing any information other than the correctness of the respective statement (this is called *zero-knowledge*).

Setup phase. The *election authority* EA determines the dates, the set of choices, the voting method, the electorate, and any other necessary data. The EA posts this information on a *public bulletin board* PBB from which all participants can read and to which they can add their messages.

Submission phase. Each eligible *voter* V_i can encrypt her individual vote v_i under the public key pk of the *tallier* T.[2] If the ballots are tallied by shuffling, then V_i encrypts her vote v_i as a single ciphertext $e_i \leftarrow \mathsf{Enc}(pk, v_i)$. If the votes are counted homomorphically, then V_i first encodes her choice v_i as a binary vector $(v_{i,l})_l$, where

[1] *Moreover, since abstention may indicate a political choice, lists of persons voting should not be published.* Article I.4, §54 of the *Code of Good Practice in Electoral Matters* of the European Commission for Democracy Through Law (Venice Commission).

[2] For simplicity, we assume that a single authority, namely T, tallies the ballots. We note that the role of the tallier can be distributed in various ways, both to reduce its trust for vote privacy and to improve robustness. Since these mechanisms are independent of DeVoS' technique, we refer to [29] for details.

$v_{i,l} = 1$ if and only if v_i is the l-th candidate, and then encrypts each bit of that representation as $e_{i,l} \leftarrow \mathsf{Enc}(\mathsf{pk}, v_{i,l})$.

In addition, V_i also computes a NIZKP π_i, the exact statement of which depends on the type of tallying. In fact, if the ballots are tallied by shuffling, then π_i (only) proves that the voter *knows* the encrypted vote. This property ensures that voters create their votes independently, which is necessary for vote privacy (see, e.g., [25]). If the ballots are tallied homomorphically, then π_i must also guarantee that the ciphertext vector $e_i = (e_{i,l})_l$ contains at most one ciphertext $e_{i,l}$ which encrypts 1 while all other ciphertexts encrypt 0; otherwise, a corrupted voter could stuff illegitimate votes or remove valid ones.

Finally, V_i authenticates to PBB and submits her ballot $b_i \leftarrow (\mathsf{V}_i, e_i, \pi_i)$.

Tallying phase. The tallier T takes all submitted ballots as input, removes possible duplicates (to protect against replay attacks [42] that violate vote privacy) and all ballots with invalid proofs, and extracts the ciphertexts $(e_i)_i$ of the remaining ballots.

If the ballots are tallied by shuffling, T first re-randomizes $(e_i)_i$ and then shuffles the result with some random permutation σ into a new ciphertext vector $(e'_{\sigma(i)})_i$. Finally, T uses the secret key sk to decrypt this ciphertext vector and posts the election result $(m'_i)_i$ on PBB. This vector is supposed to contain all the voters' choices in random order.

If the ballots are tallied homomorphically, T exploits the homomorphic property of the encryption scheme and computes a ciphertext vector $(e'_l)_l$ from $(e_{i,l})_{i,l}$, where each e'_l encrypts the number of votes for the l-th candidate. Then T uses the secret key to decrypt this ciphertext vector and posts the election result $(m'_l)_l$ on PBB. This vector is supposed to contain the total number of votes for all candidates.

In both cases, the tallier T creates and publishes a NIZKP to prove that it has processed its data correctly. In the case of shuffling, this NIZKP proves that the input ciphertexts were shuffled and decrypted correctly, whereas in the case of homomorphic aggregation, this NIZKP only proves the correctness of the final decryption, since the correctness of the homomorphic aggregation can be verified for free.

Security and trust model. The basic approaches outlined above guarantee the two most fundamental properties of secure voting: verifiability and vote privacy. For verifiability, both the voters and the tallier can be malicious, since voters can individually verify that their submitted votes are appended to the bulletin board, and everyone can verify the well-formedness of the ballots and the correctness of the tallying by checking the respective NIZKPs. Vote privacy is ensured if the tallier T is honest while all voters can be malicious, provided that all NIZKPs are indeed ZK, that the PKE scheme is IND-CPA-secure, and that the voters' NIZKP is a proof of knowledge.

Note, however, that the basic approaches do not provide any additional privacy features that are often necessary for free elections. First, voters can easily prove to a possible vote buyer or coercer how they voted. Second, everyone can see which voters participated in the election. Third, the voters' encrypted choices are public, but since perfect secrecy is impossible in a public-key setting, they could be decrypted retrospectively with the help of new cryptanalytic techniques or more powerful machines (e.g. quantum computers).

Table 1: Voting notation.

Variable	Meaning
EA	election authority
$\mathsf{V}_1, \dots, \mathsf{V}_n$	voters
v_i	V_i's vote
PT	posting trustee
T	tallier
PBB	public bulletin board
SBB	secret bulletin board

3.2 Illustration of DeVoS

We illustrate how DeVoS extends basic secure e-voting systems, as outlined in Sec. 3.1, with deniable yet verifiable vote updating and strong participation privacy.

In DeVoS, any voter can submit a new ballot that overwrites her previously submitted ballots. Since, without further means, everyone could observe which voters have re-voted and which ones have not, DeVoS hides all voters' ballots in a 'swarm' of dummy ballots. These dummy ballots do not change the voters' choices, but they are indistinguishable from the voters' real ballots. In this way, voters can secretly overwrite their votes at any time during the submission phase, providing a deniable vote update.

More specifically, DeVoS employs an additional authority called the *posting trustee* PT, which creates the cover of dummy ballots. This party is trusted for deniable vote updating and strong participation privacy, but not for (basic) vote privacy and verifiability. We note that all verifiable Internet voting protocols with strong privacy features (e.g., Civitas, BeleniosRF, or VoteAgain) necessarily use similar entities [7].

On a technical level, the following method is the key cryptographic component of DeVoS. Unlike in the basic protocols described in Sec. 3.1, for each voter V_i there exists a ballot *vector* $\vec{e}_i$ on the public bulletin board PBB, to which new ballots (e_i^j, π_i^j) are appended; initially, $\vec{e}_i$ contains a ciphertext/proof pair with "trivial" publicly known randomness and designated choice for "abstention" (e.g., $\vec{e}_i^0 = \mathsf{Enc}(\mathsf{pk}, 0; 0)$). In DeVoS, the proof π_i^j is a NIZKP of knowledge for the following disjunctive statement:

(1) If (e_i^j, π_i^j) was created by the voter V_i herself, then the proven statement guarantees that the ciphertext encrypts a choice of a valid choice and that V_i knows this choice. In particular, the ciphertext e_i^j can be completely unrelated to all previous ciphertexts $e_i^0, \dots, e_i^{j-1}$ in the ballot vector $\vec{e}_i$.
(2) Otherwise, if (e_i^j, π_i^j) was created by anyone else, the ciphertext e_i^j is a re-randomization of the previous ciphertext e_i^{j-1} in $\vec{e}_i$.

At the protocol level of DeVoS, the posting trustee PT is the authority that collects all incoming ballots and periodically updates all ballot vectors $\vec{e}_i$. For this purpose, we divide the submission

	t_0	t_1	t_2	t_3
V_1		$Enc(A)$		$Enc(B)$
V_2			$Enc(B)$	
V_3				
		↓	↓	↓
PT	$Enc(0)$	$Enc(A)$	$\xrightarrow{RE}$ $Enc(A)$	$Enc(B)$
	$Enc(0)$	$\xrightarrow{RE}$ $Enc(0)$	$Enc(B)$	$\xrightarrow{RE}$ $Enc(B)$
	$Enc(0)$	$\xrightarrow{RE}$ $Enc(0)$	$\xrightarrow{RE}$ $Enc(0)$	$\xrightarrow{RE}$ $Enc(0)$
	↓	↓	↓	↓
PBB	e_1^0	e_1^1	e_1^2	e_1^3
	e_2^0	e_2^1	e_2^2	e_2^3
	e_3^0	e_3^1	e_3^2	e_3^3

Figure 1: Submission phase of DeVoS exemplified. NIZKPs are omitted. Note that the ciphertexts (e_1^3, e_2^3, e_3^3), including two encryptions for candidate B and one encryption of 0 for abstention, are the input to the tallying phase.

phase in DeVoS into time intervals, each of which is identified by an (increasing) integer j.

Now, if V_i submits a ballot b_i^j to PT in interval j, then PT appends that ballot to $\vec{e}_i$ at the end of phase j. After the interval, V_i can individually verify whether her ballot b_i^j has been appended to PBB and then leave her virtual voting booth, guaranteeing the *vote-and-go* paradigm.

Otherwise, if V_i does not cast a vote in interval j, PT re-randomizes the current ciphertext e_i^{j-1} from $\vec{e}_i$ into e_i^j, creates a NIZKP π_i^j for the second part of the disjunction (see above), and appends this ballot $b_i^j = (e_i^j, \pi_i^j)$ to $\vec{e}_i$ at the end of phase j. These re-randomizations, which do not change the voters' choices, serve as the *dummy ballots* that collectively hide which voters re-voted and which voters participated in the election at all.

Once the submission phase is complete, the last ciphertext in V_i's vector $\vec{e}_i$, denoted by e_i, is V_i's input to the subsequent (standard) tallying phase.

The correctness of the tallying phase can be verified as in the underlying basic secure voting protocol that DeVoS extends (i.e., checking the tallier's NIZKPs).

3.3 Properties of DeVoS

We explain the main features and assumptions of DeVoS.

Remark: honest vs dishonest. Throughout this paper, we say that a party is *honest* if it follows its specified program, and *dishonest* or *corrupted* if it can run any other probabilistic polynomial time (ppt) program. Moreover, we assume that all corrupted parties are controlled by one global adversary, i.e. the most pessimistic case.

Public-key infrastructure. In DeVoS, we assume that there is a trustworthy public-key infrastructure of the voters. This is a common assumption in Internet voting systems, both in those with and without additional privacy features (e.g., BeleniosRF [5] and Belenios [12]).[3] To give a practical example, in Estonia, where the IVXV Internet voting system is used, the PKI is established with the residents' ID cards. Of course, as in any (similar) Internet voting system, we must assume that voters in DeVoS do not reveal their credentials/secret keys, since these are necessary to authenticate voters and thus to ensure that only eligible voters can vote.

Verifiability. DeVoS preserves the verifiability of the basic voting protocol that it extends. Due to the soundness of the NIZKPs π_i^j, only V_i can actually overwrite her previously cast votes, while the posting trustee PT can only append valid ballots that contain a choice for the same vote that V_i cast last. At the same time, each individual voter can verify whether her submitted ballot was appended to her ciphertext vector at the end of the respective micro submission phase. Therefore, each voter's input e_i to the tallying phase will contain V_i's actual choice, even if PT is corrupted. This means that the only new entity in DeVoS, namely PT, does not need to be trusted for verifiability either.

However, we currently do not know how to reduce the trust in the voting devices in DeVoS under realistic assumptions, while for the underlying basic secure protocols there are such methods (see, e.g., [44, 47]). The hybrid method with paper sheets in BeleniosVS [10] could provide a possible approach to solving this problem for DeVoS.

Vote privacy. DeVoS does not introduce any additional trust assumptions for vote privacy. To see this, recall that the voters' NIZKP π_i^j are proofs of knowledge, which preserves ballot independence, and that they are ZK, which guarantees that no information about the voters' choices is leaked by these proofs.

Deniable vote updating. Consider the case where a voter V_i is coerced at some point during the submission phase to cast a vote for some candidate B that the coercer prefers. In practice, this could happen if the coercer looks over the voter's shoulder and checks that she enters a vote for B and submits the resulting digital ballot. DeVoS protects against this type of coercion, assuming that the posting trustee PT is honest, since the coerced voter can later submit a ballot for her favorite choice A after the coercer has left the location. Due to the ZK property of the disjunctive NIZKP π_{Enc}, for any new ciphertext in the voter's vector $\vec{e}_i$ on the public bulletin board PBB, the coercer cannot distinguish whether this ciphertext is a new vote by the voter overwriting the coerced vote for B, or a dummy ballot by the posting trustee preserving the coerced vote.

We note that coercion-resistance in general requires that the voter cannot be monitored all the time, and therefore we need to make such an assumption for deniable vote updating in DeVoS as well. Of course, the assumption in DeVoS and related protocols such as VoteAgain [31, 41] is more specific, since it restricts the coercion-free time to the time between coercion and the end of the submission phase. Although not all voters will thus be able to use the deniable vote update of DeVoS, we believe that it can still dramatically mitigate the effect of coercion in practice.

[3] To the best of our knowledge, VoteAgain [41] is the only Internet voting protocol with strong privacy features that avoids a PKI, but this abstention was the main the reason why VoteAgain originally fell short of its security goals [31].

Strong participation privacy. If, in addition to the tallier T, the posting trustee PT is honest, then DeVoS hides which voters participate in an election and which ones do not. In fact, the ZK property of the disjunctive NIZKP π_{Enc} and the IND-CPA-security of the PKE scheme together obfuscate whether a ciphertext e_i^j is a re-randomization of the previous ciphertext e_i^{j-1} or a new encryption of a valid candidate. At the same time, if a voter V_i does not participate in an election, then her input e_i to the tallying phase encrypts a choice for abstention, since e_i is then a re-randomization of the initial ciphertext e_i^0. These two observations imply that no observer is able to distinguish whether V_i has submitted some ciphertext e_i^j during the submission phase or not.

While the reasoning above explains why DeVoS hides whether a voter who correctly follows her prescribed program has participated in an election, it also implies that a voter, who deliberately wishes to demonstrate to a coercer that she abstained from voting cannot do so convincingly. In fact, due to the soundness of π_{Enc}, there is only one designated option for all voters to choose "abstain", which precludes a coerced voter from choosing a unique invalid message that could prove her abstention. These arguments illustrate why DeVoS even provides *strong* participation privacy, which protects, among others, against forced abstention attacks.

We note that strong participation privacy in DeVoS is guaranteed under a weaker assumption on the coercion-free time window than deniable vote updating. Recall from above that for deniable vote updating we have to assume that coerced voters can secretly overwrite their coerced ballots after coercion has occurred but before the submission phase ends. For strong participation privacy, however, the coercion-free time is not restricted to the end of the submission phase, but it can be in any time window (for example at the beginning of the submission phase), which is the most minimal assumption that we can generally make for anti-coercion features.

Practical everlasting privacy. DeVoS is fully compatible with the homomorphic and the shuffling versions of the *Perfectly Private Audit Trail (PPAT)* voting protocol [15]. According to a recent systematization-of-knowledge [28], the PPAT protocols are state-of-art secure Internet voting protocols with practical everlasting privacy. In such protocols, privacy is guaranteed unconditionally towards the public, meaning that even a computationally unbounded observer would not be able to break vote privacy.

In the PPAT protocols, there is an additional secret bulletin board SBB, which is accessible only to the tallier but not to the public. The SBB contains all election data, including the ciphertexts, which remain only computationally secret due to the limitations of public-key encryption. The PBB, on the other hand, which can be read by anyone, publishes only unconditionally hiding information about the voters' choices, which is, however, sufficient to verify the correctness of the final election result. Cryptographically, PPAT uses a function to derive commitments c from ciphertexts e for the same secret messages deterministically (without knowledge of the secret key sk); analogously, the voters' NIZKP and the tallier's NIZKP can be transferred from the ciphertexts to the commitments.

Now, we essentially combine DeVoS with PPAT as follows; see Sec. 7 for full details. Instead of publishing the voters' ciphertext vectors $\vec{e}_i$ on PBB, the posting trustee PT only shares these vectors with the tallier on SBB. Cryptographically, we construct a NIZKP for the voters' disjunctive statement on the ciphertexts, from which an analogous NIZKP can be derived for the derived commitments. Using this feature, the posting trustee PT can extract from each vector $\vec{e}_i$, that is shared on SBB, a vector $\vec{c}_i$ of unconditionally hiding commitments (with associated NIZKPs), that is published on PBB. The voter V_i can then individually verify whether the derived commitment of her submitted ciphertext appears on the public bulletin board PBB, which is sufficient to ensure that her vote is actually counted.

In this way, DeVoS provides an unconditional, and therefore long-term, means of guaranteeing vote privacy, deniable vote updating and participation privacy without compromising verifiability.

Efficiency. We have implemented our PPAT instantiation of the disjunctive NIZKP π_{Enc}, the only non-standard cryptographic primitive in DeVoS. Our results show that, for example, 10,000 dummy votes can be computed per minute on a single thread on a standard laptop for 3-candidate elections; see Section 7 for further benchmarks. Since the dummy ballots can be generated by different servers/threads running in parallel, this task can easily be parallelized in practice.

In addition, the workload of the posting trustee PT can be reduced not only by distributing its role, but also by reducing the size of the dummy cover. In fact, if we implement DeVoS as illustrated in Sec. 3.2, then the posting trustee needs to create up to n dummy ballots per interval, where n is the number of voters. While such a large coverage of dummy ballots creates an ideal level of deniable vote updating, we can reduce the size of the dummy ballot cover without significantly reducing this level. First, note that dummy ballots are most effective at the end of the submission phase, so we can safely dispense with the dummy cover before that part. In Estonia, for example, voters can submit their votes online for several days. If we use DeVoS in such elections, we can limit the dummy cover to the last day or even the last hours of the election. Second, we have analyzed that DeVoS achieves a reasonable level of deniable vote updating (as measured by differential privacy [19]) even if PT creates a dummy cover for only some of those voters who did not submit a vote in a given interval (see Sec. 7.2). For example, the level of deniable vote updating remains strong even if we halve the size of the dummy cover.

This demonstrates that DeVoS can be used efficiently in practice for any size of the electorate when the computational power of the posting trustee scales linearly with the number of voters.

3.4 Threat scenario

We have summarized the security and privacy features of DeVoS in Table 2. We make the following key observations:

(1) We make no assumptions about the basic properties, public verifiability and privacy of the vote, other than those made by basic secure Internet voting systems such as Belenios [12].
(2) We make no assumptions about everlasting vote privacy other than those made by PPAT [15].
(3) We only require the honesty of the posting trustee for the additional privacy features, deniable vote updating and strong participation privacy.

Table 2: Threat scenario of DeVoS. The general assumptions on trusted parties ($\mathsf{EA} \land \mathsf{PBB}(\land \mathsf{SBB})$) and private channels ($\mathsf{EA} \rightarrow \mathsf{V}_i$) are protocol-independent and thus implicitly assumed in the following columns. $A \rightarrow B$: adversary learns when A sends a message to B, but it does not learn these messages. $A \Rightarrow B$: adversary does neither learn when A sends messages to B nor these messages. $A \leftrightarrow B$: iff $A \rightarrow B$ and $B \rightarrow A$.

	General	Public verifiability	Vote privacy	Deniable vote updating	Strong part. privacy	Everlasting vote privacy
Trusted parties	$\mathsf{EA} \land \mathsf{PBB}(\land \mathsf{SBB})$	–	T	$\mathsf{T} \land \mathsf{PT}$	$\mathsf{T} \land \mathsf{PT}$	T
Private channels	$\mathsf{EA} \rightarrow \mathsf{V}_i$	–	–	$\mathsf{V}_i \Rightarrow \mathsf{PT}$	$\mathsf{V}_i \Rightarrow \mathsf{PT}$	$\mathsf{V}_i \rightarrow \mathsf{PT} \rightarrow \mathsf{SBB} \leftrightarrow \mathsf{T}$
Coercion-free time	–	–	–	after coercion	arbitrary	–

(4) We fully trust the voters' voting devices for all security and privacy properties.

(5) We make the assumption for strong participation privacy that voters have a coercion-free time at an arbitrary point in the submission phase, while we strengthen this assumption for deniable vote updating.

Trusted parties. The third and fourth properties are equivalent to related state-of-the-art Internet voting protocols such as Civitas, BeleniosRF or VoteAgain. We do, however, not know how to distribute trust in these entities for any of these systems, including DeVoS. We also note that reducing trust in the bulletin board is usually independent of the specific voting protocol and we therefore refer to [14, 33, 35] for details, in which this problem is studied.

Regarding the fourth property, there are hybrid solutions with paper-based voting sheets, such as BeleniosVS [10], which reduce trust in the voting devices, but we are not aware of any viable solution in a fully remote environment.

Coercion-free time. We also note that in the fifth observation, the first assumption is minimal, while the second one is stronger than in fake credential approaches such as Civitas, which only require that voters are free from coercion at an arbitrary point. There is a proposal in the literature, called KTV-Helios [38], which mitigates this assumption by enabling voters to overwrite anticipated coerced votes in advance; however, in KTV-Helios, the mental work for the human voters is more complex than in DeVoS (or VoteAgain), and KTV-Helios does not protect against forced abstention attacks, while DeVoS does.

Channels. As in any secure Internet voting protocol with a voter PKI, even basic secure ones like Belenios, we assume for each feature that the voters obtain their credentials secretly from the certificate authority (in our case EA); this minimal assumption is denoted as $\mathsf{EA} \rightarrow \mathsf{V}_i$. For everlasting vote privacy, we need to additionally assume that the adversary does not learn the messages sent from the (honest) voters to the posting trustee as well as the messages exchanged between the election authorities; this minimal assumption is denoted as $\mathsf{V}_i \rightarrow \mathsf{PT} \rightarrow \mathsf{SBB} \leftrightarrow \mathsf{T}$. For (everlasting) deniable vote updating and (everlasting) strong participation privacy, we need to strengthen the assumption on the casting channel and additionally require that the adversary does not learn *when* a (coerced) voter sends a message to PT; this assumption is denoted as $\mathsf{V}_i \Rightarrow \mathsf{PT}$.

Summary. At this point we would like to summarize against which types of influence DeVoS protects and what role DeVoS therefore plays in research. Roughly speaking, DeVoS protects against 'primitive' forms of influence. Firstly, against coercers who look over voters' shoulders and check that voters also vote for the coerced candidate. As explained earlier, in DeVoS voters can only overwrite their coerced vote if they have enough time to do so before the end of the submission period. DeVoS therefore does not provide complete protection against coercion, but it does provide an efficient way to significantly reduce the impact of coercion on the final result. Furthermore, DeVoS protects against passive observers or even coercers who want to check which voters have voted based on the publicly available information on the bulletin board.

Unlike, for example, BeleniosRF, DeVoS does not provide any protection against vote buying based on cryptographic evidence. A voter who casts her vote in the last interval and is able to manipulate her voting device to output her temporary cryptographic data (i.e. the random coins r that encrypt her vote) can thus prove to an equally technically skilled vote buyer that her last ciphertext e is an encryption of v; to this end, the vote-buyer needs to check whether $e = \mathsf{Enc}(pk, v; r)$ holds true. However, as mentioned above, BeleniosRF does not protect against over-the-shoulder coercion by less technically skilled manipulators.

4 SPECIFICATION

We present the DeVoS voting protocol with full technical details.

4.1 Cryptographic primitives

We start with the cryptographic primitives used in DeVoS. Instead of relying on concrete primitives, the security of DeVoS (Sec. 5) can be guaranteed under certain assumptions the cryptographic primitives have to satisfy. We will demonstrate in Sec. 6 how to instantiate the generic DeVoS protocol efficiently.

Observe that all of the following primitives are standard in modern secure e-voting, except for the disjunctive NIZKP π_{Enc} for which we construct appropriate instantiations in Sec. 6.

Public-key encryption scheme. We use an IND-CPA-secure *public-key encryption scheme* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$.

If ballots are counted by shuffling, we assume that ciphertexts in $\mathcal{E}$ are *re-randomizable*, i.e., there exists a ppt algorithm ReRand which takes as input the public key pk and an arbitrary ciphertext $e = \mathsf{Enc}(pk, m; r)$ and outputs ciphertext e' such that $e' = \mathsf{Enc}(pk, m; r')$ for some (fresh and unknown) random r'.

If ballots are counted homomorphically, we additionally assume that ciphertexts are *additively homomorphic*, i.e., there exists a pt algorithm Sum which takes as input the public key pk, ciphertexts

$e_1 = \mathsf{Enc}(pk, m_1)$ and $e_2 = \mathsf{Enc}(pk, m_2)$, and outputs a ciphertext $e_3 = \mathsf{Enc}(\mathsf{pk}, m_3)$ that encrypts the sum $m_3 = m_1 + m_2$.

NIZKP of correct key generation. We use a NIZKP π_{KeyGen} for proving correctness of a public key pk w.r.t. $\mathcal{E}$. The underlying relation $\mathcal{R}_{\mathsf{KeyGen}}$ is $\big(x = pk, w = (r, sk)\big) \in \mathcal{R}_{\mathsf{KeyGen}} \Leftrightarrow (pk, sk) = \mathsf{KeyGen}(r)$.

One-way function. We use a *one-way function f* over $\{0,1\}^*$.

NIZKP of correct encryption. We use a NIZKP of knowledge π_{Enc} for proving correctness of a ciphertext e' w.r.t. public key pk, public verification key vk, (previous) ciphertext e, and set of valid choices C. The underlying relation $\mathcal{R}_{\mathsf{Enc}}$ is

$$\begin{aligned} &\big(x = (pk, vk, e, e', C), w\big) \in \mathcal{R}_{\mathsf{Enc}} \\ \Leftrightarrow\ &\big(w = r\colon e' = \mathsf{ReRand}(pk, e; r)\big) \vee \\ &\big(w = (ssk, m, r)\colon vk = f(ssk) \wedge e' = \mathsf{Enc}(pk, m; r) \wedge m \in C\big). \end{aligned}$$

Note that the relation $\mathcal{R}_{\mathsf{Enc}}$ is a disjunction of two statements:

(1) e' is a re-randomisation of e, or
(2) e' is a "fresh" encryption of a valid message $m \in C$ *and* the prover knows a valid secret signing key ssk for public verification key vk.

NIZKP or correct tallying. Since the tallying method of the underlying basic secure e-voting system does not need to be modified, we do not specify the NIZKPs that are used in the tallying phase, but refer to [29].

4.2 Protocol participants

The DeVoS protocol is run among the following participants: Election authority EA, public bulletin board PBB, voters $\mathsf{V}_1, \ldots, \mathsf{V}_n$, tallier T, and posting trustee PT. Observe that all participants except for the posting trustee PT are standard (see Sec. 3.1). In order to simplify the presentation of DeVoS, we assume that the respective programs by T and PT are run by single participants, noting that their roles can be distributed using standard techniques.

We assume that all messages from the officials/trustees (i.e., EA, T, and PT) on the public bulletin board PBB are authenticated; this could be achieved by using digital signatures. We also assume that voters implicitly authenticate themselves to the posting trustee PT, and that PT only accepts ballots from the respective authenticated voters. Since the exact method of authentication is not relevant to the overall protocol, we abstract away from it here.

4.3 Setup phase

We describe the honest programs run in the setup phase of DeVoS.

Election authority. The election authority EA determines all election parameters and posts them to the bulletin board: security parameter 1^ℓ, list of eligible voters $\vec{\mathsf{id}}$, *micro submission periods* $t_0, t_1, \ldots, t_e$ (t_0: starting time, t_e: end time of submission period), election ID $\mathsf{id}_{\mathsf{election}}$, and the set of valid choices C.

For each voter V_i ($i \in \vec{\mathsf{id}}$), the election authority EA chooses a secret "signing" key $ssk_i \leftarrow \{0,1\}^\ell$ uniformly at random and computes the corresponding public verification key as $vk_i \leftarrow f(ssk_i)$, where f is the one-way function mentioned above. The authority publishes the list of public verification keys $(vvk_i)_{i \in \vec{\mathsf{id}}}$ on PBB and sends the credentials ssk_i to each voter.[4] Finally, EA initializes each voter's ballot vector on PBB as $\vec{e}_i = \big(\mathsf{Enc}(pk, 0; 0), \epsilon\big)$.

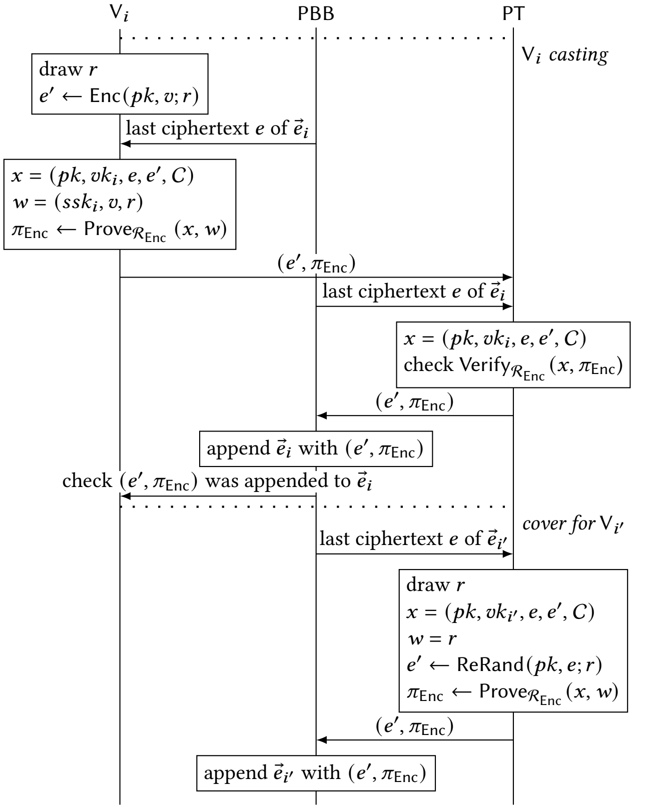
V_i
PBB
PT
V_i *casting*
draw r
$e' \leftarrow \mathsf{Enc}(pk, v; r)$
last ciphertext e of $\vec{e}_i$
$x = (pk, vk_i, e, e', C)$
$w = (ssk_i, v, r)$
$\pi_{\mathsf{Enc}} \leftarrow \mathsf{Prove}_{\mathcal{R}_{\mathsf{Enc}}}(x, w)$
(e', π_{Enc})
last ciphertext e of $\vec{e}_i$
$x = (pk, vk_i, e, e', C)$
check $\mathsf{Verify}_{\mathcal{R}_{\mathsf{Enc}}}(x, \pi_{\mathsf{Enc}})$
(e', π_{Enc})
append $\vec{e}_i$ with (e', π_{Enc})
check (e', π_{Enc}) was appended to $\vec{e}_i$
cover for $\mathsf{V}_{i'}$
last ciphertext e of $\vec{e}_{i'}$
draw r
$x = (pk, vk_{i'}, e, e', C)$
$w = r$
$e' \leftarrow \mathsf{ReRand}(pk, e; r)$
$\pi_{\mathsf{Enc}} \leftarrow \mathsf{Prove}_{\mathcal{R}_{\mathsf{Enc}}}(x, w)$
(e', π_{Enc})
append $\vec{e}_{i'}$ with (e', π_{Enc})

Figure 2: Submission phase of DeVoS for voter V_i who voted and voter $\mathsf{V}_{i'}$ who abstained.

Decryption trustee. The decryption trustee T runs the key generation algorithm of the public-key encryption scheme $\mathcal{E}$ to generate its public/private (encryption/decryption) key pair (pk, sk). In addition, T creates a NIZKP π_{KeyGen} to prove the validity of pk and posts $(pk, \pi_{\mathsf{KeyGen}})$ to the public bulletin board PBB.

4.4 Submission phase

We describe the honest programs run in the submission phase.

Voter. Voter V_i runs the following program:

(1) Pick favorite choice $v \in C$ and encrypt it under pk to obtain $e' \leftarrow \mathsf{Enc}(pk, v; r)$.
(2) Read the latest status of vector $\vec{e}_i$, including the currently last ciphertext e in $\vec{e}_i$, from the bulletin board PBB.
(3) Use ssk_i, vote v, and randomness r to create a NIZKPoK π_{Enc} for the tuple (pk, vk_i, e, e', C).[5]
(4) Send (e', π_{Enc}) to the posting trustee PT.

[4] For the sake of transparency, we decided to explicitly create the PKI among the voters in our formal model of DeVoS. In real practical elections, existing PKIs could be used instead, such as national digital identities, as is the case in Estonia.

[5] Observe that V_i proves the right term of the disjunctive relation $\mathcal{R}_{\mathsf{Enc}}$ (see above), i.e., that e' is a "fresh" encryption of a valid vote $v \in C$ *and* the voter (prover) knows a valid secret signing key ssk_i for public verification key vk_i.

Then, analogous to the basic voting protocol (see Sec. 3.1), the voter V_i can individually verify whether her ballot (e', π_{Enc}) is contained in $\vec{e}_i$ as published by PT on PBB. Notably, the voter can perform this check immediately after the current micro submission phase ends and the voter exits the virtual booth. If $\vec{e}_i$ does not contain (e', π_{Enc}), then the voter can post a complaint to PBB.

Posting trustee. Posting trustee PT runs the following program (in a given micro submission period t_l):

(1) If voter V_i sends ballot (e', π_{Enc}) then:
  (a) Ignore ballot if π_{Enc} is not valid w.r.t. vk_i, pk, e, where e is latest ciphertext in $\vec{e}_i$, or if V_i has already sent a valid ballot in t_l.
  (b) Append (e', π_{Enc}) to $\vec{e}_i$ on PBB otherwise.
(2) If voter V_i has not sent valid ballot during t_l then:
  (a) Compute $e' \leftarrow \mathsf{ReRand}(pk, e; r)$, where e is latest ciphertext in $\vec{e}_i$.
  (b) Use r to compute NIZKPoK π_{Enc} for the tuple (pk, vk_i, e, e', C).[6]
  (c) Append (e', π_{Enc}) to $\vec{e}_i$ on PBB.

At the end of the micro-submission period, the posting trustee PT updates all ballot vectors $\vec{e}_i$ on the public bulletin board PBB.

4.5 Tallying and verification phases

The tallying phase of DeVoS is standard: either verifiable shuffling [29] or homomorphic aggregation with subsequent verifiable decryption. We denote the NIZKP by the tallier do prove the correctness of the tallying phase by π_{T}.

Any participant, including the voters or external observers, can verify the correctness of the previous phases, essentially by checking the correctness of all NIZKPs published during the setup, submission, and tallying phase.

5 SECURITY

We present the main results of DeVoS' security in terms of verifiability, vote privacy, deniable vote updating, and strong participation privacy. Concerning the last two, we also provide differential privacy results.

5.1 Verifiability

We present the verifiability result of DeVoS.

Definition. An e-voting protocol is *verifiable* if the final election result is accepted only if it corresponds to the actual choices of the voters. We follow [11, 39] to formalize verifiability as follows.

The verifiability definition [11, 39] is centered around a "virtual" entity, called the *judge* J. In reality, the program of the judge can be executed by any party, including external observers and even voters themselves. In a given protocol run, the judge J takes as input solely public information (e.g., the zero-knowledge proofs in DeVoS published on PBB) and then performs certain checks.

Specifically, the judge J in DeVoS performs the following checks and accepts a protocol run if all of them pass, otherwise it rejects it. The judge reads from PBB the data published by the election authority during the setup phase. The judge then uses this information to verify all the NIZKPs published on PBB during the setup, submission, and tallying phase. The judge also checks to see if any voter has posted a complaint on PBB that their submitted ballot was not included. Note that the input to the judge in DeVoS is only public data from PBB.

Result. We make the following assumptions for verifiability:

(V1) The PKE scheme $\mathcal{E}$ with re-randomization is correct (for verifiability, IND-CPA-security is not needed).
(V2) The function $f\colon \{0,1\}^* \rightarrow \{0,1\}^*$ is one-way.
(V3) $\pi_{\mathsf{KeyGen}} and \pi_{\mathsf{T}}$ are NIP systems (for verifiability, ZK is not needed) and π_{Enc} is a NIZKP.
(V4) The voting authority EA, the public bulletin board PBB, and the judge J are honest.
(V5) The messages from EA to any V_i remain private.

Note that the trust assumptions (V4) and the assumption that voters obtain their voting credentials privately (V5) are the same as in basic secure e-voting protocols. The other assumptions, (V1) to (V3), can be realized with appropriate cryptographic building blocks (see Sec. 6).

We now state the verifiability result of DeVoS. We present and prove the formal theorem in Appendix B.

THEOREM 5.1 (VERIFIABILITY (INFORMAL)). *Under the assumptions (V1) to (V5) specified above, DeVoS is verifiable by judge* J.

Verifiability essentially follows from the global relation that is implied by the local relations of all individual NIZKPs published on PBB and the fact that each individual voter can verify whether the ballot she submitted has been appended to her ciphertext vector at the end of the respective micro submission phase.

Remark. We note that DeVoS shares the following drawback with all e-voting protocols that offer strong privacy features (e.g., BeleniosRF, Civitas, KTV-Helios); according to [7], this property is intrinsic to such protocols. If the party (in our case EA) that provides the voters' public-key infrastructure and the additional party (in our case PT) that ensures the strong privacy features are corrupt and collude, then they can manipulate the election result undetected. In practice, it is therefore important to make sure that these two agents are run by different entities whose trustworthiness is carefully checked (by non-cryptographic means).

5.2 Vote privacy

We present the vote privacy result of DeVoS.

Definition. An e-voting protocol provides *vote privacy* if all data published during an election does not reveal more information about individual voters' choices than what can be deduced from the final election result.

We follow [40] to formalize vote privacy as follows. Their definition measures the *privacy loss* of a voting protocol as the advantage of an observer in distinguishing whether an arbitrary honest voter V_{obs}, called the *voter under observation*, has voted for the valid choice m_0 or the valid choice m_1. In short, we say that a voting protocol offers *vote privacy* if, for any voter under observation V_{obs}, this advantage is negligibly close to the advantage of any observer in an ideal voting protocol that merely outputs the final result.

[6] Observe that PT proves the left term of the disjunctive relation $\mathcal{R}_{\mathsf{Enc}}$ (see above), i.e., that e' is a re-encryption of e *and* PT knows the randomness used to re-encrypt e.

Result. We make the following assumptions for vote privacy:

(VP1) The PKE scheme with re-randomization is IND-CPA-secure.
(VP2) The function $f\colon \{0,1\}^* \rightarrow \{0,1\}^*$ is one-way.
(VP3) $\pi_{\mathsf{KeyGen}}, \pi_{\mathsf{T}}$ are NIZKPs and π_{Enc} is a NIZKPoK.
(VP4) The voting authority EA, the public bulletin board PBB, and the tallier T are honest.
(VP5) The messages from EA to any V_i remain private.

Note that the trust assumptions (V4) and the assumption that voters obtain their voting credentials privately (V5) are the same as in basic secure e-voting protocols. The other assumptions, (V1) to (V3), can be realized with appropriate cryptographic building blocks (see Sec. 6).

We now state the vote privacy result of DeVoS. We present and prove the formal theorem in Appendix C.

Theorem 5.2 (Vote privacy (informal)). *Under the assumptions (VP1) to (VP5) specified above, DeVoS offers vote privacy.*

Vote privacy is essentially a consequence of the privacy of the basic voting protocol, which DeVoS extends.

5.3 Deniable vote updating

We state the deniable vote updating result of DeVoS.

Definition. An e-voting protocol provides *deniable vote updating* if all data published during an election hide which voters updated their potentially coerced votes during the submission phase and which ones did not. We formalize this property as a specific instance of the generic definition of coercion-resistance proposed in [40].

The coercion-resistance definition of [40] requires that each coerced voter has the option of executing a counter-strategy that (1) allows the voter to achieve her own goal, but (2) is indistinguishable from the program the voter would execute if she obeyed the coercer. In the case of deniable vote updating, we consider coercers who want to voters to submit a particular ballot (e.g., a vote for B), while the voters want to overwrite their previously submitted coerced ballots (e.g., with a vote for A). The concrete counter-strategy that these voters can use in DeVoS is trivial: they simply submit a new ballot for their favorite candidate.

Assumptions. For deniable vote updating of DeVoS, the first three assumptions (DVU1) to (DVU3) are equal to (VP1) to (VP3) (see above), and we assume:

(DVU4) The voting authority EA, the public bulletin board PBB, the tallier T, *and the posting trustee* PT are honest.
(DVU5) The messages from any V_i to PT and their times remain private.
(DVU6) For each coerced voter, there exists a micro submission phase after the coercer's supervision.

The additional assumptions (DVU4) to (DVU6) are essentially standard in coercion-resistant e-voting protocols, such as Civitas [9] or VoteAgain [41]. First, as specified in (DVU4), they require an entity that is trusted to protect against coercion (in our case the posting trustee PT). Second, they assume that voters cannot be coerced during the entire submission phase: in our case, we assume that a coerced voter can submit a new ballot after the coercer has left (DVU6). We believe that this assumption is reasonable for a significant part of the electorate in ordinary real elections, which means that the impact of coercion can be mitigated significantly with our technique for such elections.

Result. We now state the deniable vote updating theorem of DeVoS. We present and prove the formal result in Appendix D.

Theorem 5.3 (Deniable vote updating (informal)). *Under the assumptions (DVU1) to (DVU6) specified above, DeVoS offers deniable vote updating.*

5.4 Strong participation privacy

We state the strong participation privacy result of DeVoS.

Definition. An e-voting protocol provides *participation privacy* if all data published during an election hide which *honest* voters participated in the election and which ones did not. This notion can be formalized directly with the vote privacy definition of [40] (see above) by considering "abstention" as a valid choice.

We say that an e-voting protocol achieves *strong participation privacy* if it provides participation privacy and also guarantees that voters cannot prove convincingly that they abstained. We formalize the latter property as a specific instance of the generic definition of coercion-resistance proposed in [40] as follows.

Recall from above that the coercion-resistance definition of [40] requires that each coerced voter has the option of executing a counter-strategy that (1) allows the voter to achieve her own goal, but (2) is indistinguishable from the program the voter would execute if she obeyed the coercer. In the case of strong participation privacy, we consider voters whose goal is to participate in the election, while the coercer wants them to abstain from voting. The concrete counter-strategy that these voters can use in DeVoS is trivial: they simply submit a ballot.

Assumptions. For strong participation privacy of DeVoS, the assumptions (PP1) to (PP6) are the same as (DVU1) to (DVU6) that we made for deniable vote updating (see above). Again, we note that these assumptions are essentially standard in such voting protocols.

Result. We now state the strong participation privacy theorem of DeVoS. We present and prove the formal result in Appendix D.

Theorem 5.4 (Strong participation privacy (informal)). *Under the assumptions (PP1) to (PP6) specified above, DeVoS offers strong participation privacy.*

Essentially, DeVoS offers strong participation privacy due to PT's dummy ballot cover, as well as the voters' NIZKP π_{Enc} which ensures validity of the votes and thus protects against coercers who demand that voters submit unique invalid messages.

6 INSTANTIATIONS

We propose two concrete instantiations of the abstract cryptographic primitives of DeVoS (Sec. 4.1). Our first instantiation is compatible with ElGamal-based Internet voting protocols, such as Belenios [12], the most widely used secure Internet voting systems in practice. Our second instantiation DeVoS^{EL} is compatible with *Perfectly Private Audit Trail (PPAT)* [15], the state-of-the-art verifiable e-voting framework with practical everlasting privacy.

Due to space limitations, we focus on DeVoS^{EL} in this section, and elaborate on the instantiation with conditional public privacy

$$\begin{tikzcd} \text{SBB :} & \mathsf{e}_i^{j-1} \arrow[r, "\pi_{\mathsf{Enc}}"] \arrow[d, "\mathsf{DeriveCom}^{\ddagger}"] & \mathsf{e}_i^{j} \arrow[d, "\mathsf{DeriveCom}^{\ddagger}"] \\ \text{PBB :} & \mathsf{c}_i^{j-1} \arrow[r, "\pi_{\mathsf{Com}}"] & \mathsf{c}_i^{j} \end{tikzcd}$$

Figure 3: NIZKP of well-formedness in DeVoS^{EL}.

in Appendix E. In Sec. 7, we will show that both instantiations are practically efficient.

Idea. At a high level, in DeVoS^{EL}, we separate the data shared on the bulletin board as follows. The public part of the bulletin board, called PBB, contains all the information needed to verify the correctness of the tallying procedure and thus the final result. This part of the bulletin board is accessible to all parties in order to maintain public verifiability. The secret part of the bulletin board, called SBB, contains all the information to tally the voters' ballots. This section can only be accessed by the authorities/trustees EA, PT and T, but not by the voters or external observers.

Now, the secret bulletin board SBB in DeVoS^{EL} contains the same information as the public bulletin board PBB in DeVoS, while PBB in DeVoS^{EL} contains only unconditionally hidden information about the voters' choices. In particular, just like PPAT, DeVoS^{EL} uses commitment-consistent encryption (CCE) [15], a notion of PKE that allows anyone to deterministically derive an unconditionally hiding commitment to a message from a ciphertext to the same message, without knowledge of the secret key. Unlike the ciphertexts on SBB, which can eventually be decrypted by powerful adversaries, the commitments on PBB reveal no information even to computationally unbounded adversaries. Analogously, it is possible to deterministically derive NIZKPs with unconditional ZK that prove relations between the commitments on PBB from the NIZKPs (with unconditional ZK) that prove the same relations between the corresponding ciphertexts on SBB. We illustrate this concept in Fig. 3.

The different phases of DeVoS can now easily be updated: whenever a party in the original DeVoS protocol sends a message to the bulletin board, the bulletin board posts this message to the secret section SBB and uses the CCE property to derive the corresponding (unconditionally secret) message and posts it to PBB.

Cryptographic realization. We realize the abstract CCE scheme with an original instantiation from [15], which can be seen as an extended version of ElGamal PKE, from which Pedersen commitments can be derived. In the remainder of this section, we recall this concrete primitive and explain how to build the non-standard NIZKP π_{Enc} on top of it.

The NIZKP of correct key generation π_{KeyGen} is (again) trivial. For the NIZKP of shuffle π_{Shuffle}, we refer to [26], which is a machine-checked variant of [52]. For the NIZKP of correct decryption, we refer to [15].

The public verifiability proof of PPAT (see Appendix B) carries over to DeVoS^{EL}, since it is implied by the individual NIZKPs on PBB, whose respective relations remain the equivalent. The practical everlasting privacy of PPAT follows from the fact that all information (commitments and NIZKPs) on PBB is unconditionally secret (hiding or ZK, respectively).

Commitment-consistent encryption. A commitment consistent encryption scheme $\mathcal{E}^{\ddagger}$ is a tuple of PPT algorithms ($\mathsf{Gen}^{\ddagger}$, $\mathsf{KeyGen}^{\ddagger}$, $\mathsf{Enc}^{\ddagger}$, $\mathsf{Dec}^{\ddagger}$, $\mathsf{DeriveCom}^{\ddagger}$, $\mathsf{Open}^{\ddagger}$, $\mathsf{Verify}^{\ddagger}$), where $\mathsf{Gen}^{\ddagger}$ outputs public parameters pp on input a security parameter, ($\mathsf{KeyGen}^{\ddagger}$, $\mathsf{Enc}^{\ddagger}$, $\mathsf{Dec}^{\ddagger}$) is a PKE scheme, and ($\mathsf{DeriveCom}^{\ddagger}$, $\mathsf{Open}^{\ddagger}$, $\mathsf{Verify}^{\ddagger}$) are defined as follows:

- $\mathsf{DeriveCom}^{\ddagger}(pk, e)$: takes a ciphertext e as input and outputs a commitment c using pk.
- $\mathsf{Open}^{\ddagger}(sk, e)$: takes a ciphertext e as input and outputs an auxiliary value a, that can be considered as part of an opening for a commitment c.
- $\mathsf{Verify}^{\ddagger}(pk, c, m, a)$: takes a message m, a commitment c w.r.t. public key pk, and an auxiliary value a as inputs and outputs a bit. The algorithm checks if the opening (m, a) is valid w.r.t. c and pk.

The following correctness property should be satisfied. For any triple (pp, pk, sk) output by $\mathsf{Gen}^{\ddagger}$ and $\mathsf{KeyGen}^{\ddagger}$, any message m from the message space and any $e = \mathsf{Enc}^{\ddagger}(pk, m)$, it holds with overwhelming probability in the security parameter that $\mathsf{Dec}^{\ddagger}(sk, e) = m$ and

$$\mathsf{Verify}^{\ddagger}(pk, \mathsf{DeriveCom}^{\ddagger}(pk, e), \mathsf{Dec}^{\ddagger}(sk, e), \mathsf{Open}^{\ddagger}(sk, e)) = 1.$$

Similarly to the CPA-secure encryption scheme used in DeVoS, we assume that it is possible to efficiently re-randomize ciphertext $e = \mathsf{Enc}^{\ddagger}(pk, m; r)$ into $e' = \mathsf{Enc}^{\ddagger}(pk, m; r')$ using pk and some "fresh" randomness. Moreover, we assume that any commitment $c = \mathsf{DeriveCom}^{\ddagger}(pk, \mathsf{Enc}^{\ddagger}(pk, m; r))$ can be efficiently re-randomized into $c' = \mathsf{DeriveCom}^{\ddagger}(pk, \mathsf{Enc}^{\ddagger}(pk, m; r'))$ using pk and some "fresh" randomness.

We now describe the exponential CCE encryption scheme instantiated from bilinear groups, which is a slight variant of [15]. Let $\Lambda = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g, h)$ be a description of bilinear groups, where g is a generator of $\mathbb{G}_1$, h is a generator of $\mathbb{G}_2$, and $\hat{e}$ is an efficient and non-degenerate bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. We assume the DDH problem is hard in both $\mathbb{G}_1$ and $\mathbb{G}_2$. The scheme is defined as follows:

- $\mathsf{Gen}^{\ddagger}(\lambda)$: on input security parameter λ generate $\Lambda = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, g, h)$, choose $h_1 \in \mathbb{G}_2$, and set $pp = (\Lambda, h_1)$. The public parameter pp is implicitly given as input to the rest of algorithms.
- $\mathsf{KeyGen}^{\ddagger}()$: choose $(x_1, x_2) \leftarrow_{\$} \mathbb{Z}_q^2$, set $g_1 = g^{x_1}, g_2 = g^{x_2} \in \mathbb{G}_1$, output $(sk = (x_1, x_2), pk = (g_1, g_2))$.
- $\mathsf{Enc}^{\ddagger}(pk, m)$: choose $(r_1, r_2, r_3) \leftarrow_{\$} \mathbb{Z}_q^3$, compute $(e_1, e_2, e_3, c_1, c_2) \leftarrow (g^{r_2}, g^{r_3}, g_1^{r_1} g_2^{r_3}, h^{r_1} h_1^{r_2}, g_1^{r_2} g^m)$, output $e = (e_1, e_2, e_3, c_1, c_2)$.
- $\mathsf{Dec}^{\ddagger}(sk, e)$: return $\log_g(c_2 / e_1^{x_1})$.
- $\mathsf{DeriveCom}^{\ddagger}(pk, e)$: return (c_1, c_2).
- $\mathsf{Open}^{\ddagger}(sk, e)$: return $(a = e_3 / e_2^{x_2})$.
- $\mathsf{Verify}^{\ddagger}(pk, c, m, a)$: return 1 if $\hat{e}(g, c_1) = \hat{e}(a, h)\hat{e}(c_2 / g^m, h_1)$.

We refer to App. F for details on the NIZKP of well-formedness that we constructed.

7 EFFICIENCY

We study the practical efficiency of DeVoS and different optimization strategies. First, we show how the abstract cryptographic primitives employed in DeVoS (Section 6) can be instantiated and that the resulting system can be implemented efficiently. Afterwards, we analyze different approaches to significantly reduce the workload of the posting trustee while maintaining a reasonable level of deniable vote updating and strong participation privacy.

7.1 Implementation

We implemented a proof-of-concept prototype[7] of the NIZKPoK of correct encryption π_{Enc} because this primitive is the only non-standard primitive in our instantiation of DeVoS, and because the posting trustee PT needs to compute this proof numerous times during the submission phase.

We use the relic C-library [1] as a backend, relicwrapper [3] for binding relic with C++/python, and the boost library. The relic library is configured to use the gmp library for arithmetic operations and makes use of several optimization techniques by default, such as Comba multiplication, Montgomery modular reduction, and sliding window modular exponentiation.

Experiments. All computations are done locally in a single thread on an eight-core Intel Core i7-8565U 1.80Ghz Linux machine with 16G RAM and 1TB SSD. We start by measuring the running time of the proof and verification algorithms of the π_{Enc} proof system in DeVoS. For the $m \in C$ part of the proof system in a shuffle-based tallying, we first instantiate classical range proofs for the statement $m \in [0, 2^{\tau} - 1]$, which boils down to proving τ times that an ElGamal ciphertext encrypts either 0 or 1 [6, 13]. If the votes are tallied homomorphically, τ ElGamal ciphertexts encrypting either 0 or 1, allow to encode $|C| = \tau - 1$ choices. To simplify the benchmarks, in both tallying methods we measure the runtime of the encryption and re-randomisation algorithms of arbitrary τ bits; the π_{Enc} part is not affected by this simplification. We refer to Tab. 3 (the left part) for the running time of the proof system π_{Enc} in DeVoS.

We can instantiate $m \in C$ more efficiently, e.g., using log-proofs [17]. We adapt the reference implementation [17] for log-proofs from the numerical setting to bilinear maps and measure the running time of solely the $m \in C$ part. We compare the log-proofs with the classical approach used in the first part of the experiment. The results are averaged per 10 runs. We refer to Tab. 3 (the right part) for the respective micro benchmark numbers.

Analysis. The proof system uses an OR-proof of the re-randomisation (dummy ballots) and new encryptions (fresh votes by the voters) together the proof of correct choice, which depends on the number of vote choices. We note that adding the well-formedness component ($m \in C$) to the NIZK proofs does not significantly affect the overall performance. If a shuffle-based tallying is used, instantiating NIZK proofs of well-formedness using range proofs incurs only a logarithmic overhead, whereas this component is essentially free if the votes are tallied homomorphically.

Table 3: Experimental results for proving $\mathcal{R}_{\mathsf{Enc}}$ in DeVoS depending on τ, in milliseconds. For the mix tallying $|C| = 2^{\tau}$, for the homomorphic tallying $|C| = \tau - 1$. On the left part, we report the running time of computing new ciphertext, proving the relation, and verifying the relation. On the right part, we only compare the cost of range proofs, instantiated classically or via log-proofs.

τ	fresh vote re-rand.	prove	verify	clas.	log
2	2.31±0.87	8.09±2.78	11.66±2.74	4.49±0.85	3.64±0.85
	2.00±0.66	9.47±1.16	11.48±2.74		
4	3.55±1.97	15.87±3.31	21.93±2.26	8.06±1.02	5.37±1.14
	3.73±0.94	17.56±1.48	20.73±1.34		
8	6.41±0.70	29.64±4.63	42.89±6.66	15.64±1.34	7.14±1.12
	7.23±2.60	35.98±8.12	42.34±8.37		
32	25.94±6.15	115.07±6.71	176.68±15.38	61.61±0.72	12.36±0.82
	30.85±9.27	156.06±7.80	176.91±12.40		
128	102.08±8.61	487.85±31.35	711.81±34.38	246.22±8.34	22.01±1.89
	104.48±15.35	594.37±5.89	703.77±36.08		
512	392.52±5.64	1913.51±243.79	2745.04±299.28	981.23±11.71	55.88±3.03
	437.24±24.80	2262.24±227.46	2563.39±232.39		

For a small parameter such as $\tau = 2$, which corresponds to four choices in the shuffle-based version and one choice in the homomorphic version, both computing a proof for a pair of ciphertexts and verifying the proof takes less than 12 ms, respectively.

When running in a single thread, the posting trustee can generate 10,000 votes in about 2 minutes. Note that our system is highly parallelizable, as the ballots of different voters are not related to each other. Together with the optimization, a single posting trustee can take care of 20,000 voters if each micro-submission phase lasts 2 minutes and only for half of the voters dummy ballots are added (see Section 5); this number increases to 80,000 when running on (perfect) 4 threads. The ciphertext size is 2 group elements, the encryption proof size depends on a concrete instantiation of range proofs.

We observe that, since the cost of the proof is dominated by the $|C|$ term, we can upgrade DeVoS to DeVoS^{EL} at little extra cost.

7.2 Optimizations

We study different ways to improve the efficiency of DeVoS by reducing the size of the dummy cover. Specifically, we analyze the level of deniable vote updating in the following four strategies:

(1) PT generates ballots only in the last k submission phases.

(2) The voters are divided into k' groups (e.g., $\{1, 2, \ldots, k'\}$), and in the ith submission phase PT adds dummy ballots only to those voters who are in the group $i \pmod{k'}$ and do not submit a ballot during the current submission phase.[8]

(3) PT adds dummy ballots with probability $p < 1$ for those voters who do not vote during the current submission phase.

(4) PT adds dummy ballots to some (randomly selected) voters who do not submit a ballot during the current submission phase, so that there is a ballot (real or dummy) for exactly a fraction $p' < 1$ of the voters.[9]

[7] https://gitlab.uni.lu/APSIA/DeVoS

[8] Here we have assumed $m \equiv 0 \pmod{k'}$.

[9] Here we have assume that less than a fraction p' of voters vote in a submission phase.

Reducing the scope of the deniability guarantee. Strategies (1) and (2) reduce the posting trustee's workload by restricting the dummy cover to certain time windows. Hence, any vote update outside of these windows will not be deniable. Strategy (1) reduces the workload to a fraction of $\frac{k}{m}$, but provides only a single window for deniable vote updates. In contrast, due to its periodic nature, strategy (2) provides $\frac{m}{k'}$ distinct opportunities for deniable vote updates while reducing the cost to $\frac{1}{k'}$ fraction.

Relaxing the strength of the deniability guarantee. Another way to reduce the (computational and storage) burden on PT is to relax the deniable voting guarantee rather than reduce its range. This way, providing the same (but relaxed) is provided throughout the entire submission phase. This is the goal of strategy (3), which reduces the expected work load of PT to a fraction of p. Strategy (4) improves on this probabilistic efficiency improvement by fixing the number of ballots (dummy or legit) within a micro submission phase. In the following, we analyze these strategies using *Differential Privacy*.

Differential Privacy Relaxations. We assume the reader is familiar with differential privacy (DP) [19]; a short introduction is presented in App. G. In a nutshell, by adding noise, DP ensures that an arbitrary change on a single record in the database has a small (bounded) effect on the result. DP could be adopted to DeVoS quite naturally: the data set can be the state of the bulletin board, so the records are the votes (i.e., DP protects votes instead of voters).

If only a limited number of dummy ballots are created, this 'positive' noise makes the attacker uncertain about the validity of a recorded ballot. Without 'negative' noise (i.e. removing real votes and not dummy ones), the protection is inherently asymmetric, since the attacker can be sure that a missing ballot means that no vote was cast. Thus, there is a trade-off in such optimization: higher efficiency comes at the cost of weaker privacy protection.

Indeed, in DeVoS, the PT can only add dummy votes (and hide the presence of a ballot), but cannot remove valid ballots (to hide the absence of a ballot), as this would directly contradict the correctness of the final result. This asymmetry could be built directly into the definition, as in [37, 51]. In this paper we use *one-sided DP* (OSDP) [37], which only protects the sensitive state defined by policy P. In our case, the two states are the sensitive 'presence' and the non-sensitive 'absence'. Note that OSDP does not solve the privacy leakage of non-existent votes; our goal is to show what privacy guarantee DeVoS still achieves when optimized, i.e. what is the price (in privacy) of the increased efficiency.

DP analysis. First, we focus on a single submission period and show in Theorem 7.1 that strategy (3) satisfies one-sided DP with the parameter $\epsilon = -\log p$. The proof of this theorem and the result for strategy (4) are given in Appendix G.

Theorem 7.1. *If* PT *generates each dummy ballot for each voter according to the IID* Bernoulli(p) *distribution with* $p < 1$, *then strategy (3) satisfies* $\log\left(\frac{1}{p}\right)$*-OSDP within a submission phase where the presence of a ballot is protected.*

Extending the analysis to multiple rounds is straightforward thanks to the composition property of DP [16]: for instance, if strategy (3) yields $\log\frac{1}{p}$-OSDP in a single micro submission phase, then it yields $c \cdot \log\frac{1}{p}$-OSDP for c micro submission phases. Since this grows linearly with c, the DP guarantee deteriorates rapidly. In fact, DP is a worst-case guarantee, i.e. it covers cases where voters update their votes in every submission phase. Without protecting against such outliers, it is possible to have a tighter DP guarantee, as Theorem 7.2 shows for strategy (3). We present our proof in Appendix G.

Theorem 7.2. *If* PT *generates each dummy ballot for each voter according to the IID* Bernoulli(p) *distribution with* $p < 1$ *(i.e., via strategy (3)), and 'after a coercion' the voters update their votes at most c times*[10]*, then regardless of the number of remaining micro submission phases, their votes enjoy* $c \cdot \log\frac{1}{p}$*-OSDP.*

Recall from Section 3 that the main goal of DeVoS is to offer deniable vote updating and strong participation privacy. Therefore, assuming that each voter updates her vote at most once, the privacy parameter for OSDP that strategy (3) provides for the entire protocol run is $\log p^{-1}$. Note that this is independent of the remaining rounds, i.e., of the micro submission phase in which coercion took place. As a best practice, the rule of thumb is to set the privacy parameter of any DP mechanism below 1.0. Otherwise, the privacy guarantee provided may be too weak. As a consequence, strategy (3) with $p > \frac{1}{e} \approx 0.37$ could still imply more than 60% efficiency improvement. Finally, despite this strong result, it is important to note that DP provides a probabilistic guarantee. Thus, it is still possible (to some extent based on the privacy parameter) to use statistical tests (similar to [50]) to determine whether a vote has been updated.

8 FUTURE WORK

We plan to extend our initial contribution in three aspects. First, we plan to analyze the usability of DeVoS and related Internet voting protocols in order to learn which techniques are effective in practice; we conjecture that the voter ceremony in DeVoS is sufficiently intuitive to offer its strong privacy features not only in theory but also in real-world elections. Second, we intend to study how trust in the voters' voting devices in DeVoS can be mitigated, for example by augmenting the protocol with ballot sheets as in BeleniosVS. Third, we plan to complement our pen-and-paper cryptographic security analysis of DeVoS with computer-aided verification tools.

ACKNOWLEDGMENTS

We thank the anonymous PETS reviewers for their constructive and detailed feedback, which helped to improve the overall quality of the paper. We also thank Najmeh Soroush for discussions of an earlier version of the DeVoS protocol.

Johannes Müller and Ivan Pryvalov were supported by the Luxembourg National Research Fund (FNR), under the CORE Junior project FP2 (C20/IS/14698166/FP2 /Mueller). Balazs Pejo was supported by the Ministry of Innovation and Technology from the NRDI Fund, financed under the *FK*_21 funding scheme (project no. 138903). Moreover, Ivan Pryvalov was supported by the German Federal Government, the Federal Ministry of Education and Research and the State of Brandenburg within the framework of

[10] We assume that the votes are uniformly distributed in the remaining submission phases.

the joint project EIZ: Energy Innovation Center (project numbers 85056897 and 03SF0693A) with funds from the Structural Development Act (Strukturstärkungsgesetz) for coal-mining regions.

REFERENCES

[1] D. F. Aranha, C. P. L. Gouvêa, T. Markmann, R. S. Wahby, and K. Liao. 2023. RELIC is an Efficient LIbrary for Cryptography. Retrieved May 31, 2023 from https://github.com/relic-toolkit/relic

[2] David Bernhard, Olivier Pereira, and Bogdan Warinschi. 2012. How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Proceedings (Lecture Notes in Computer Science, Vol. 7658)*, Xiaoyun Wang and Kazue Sako (Eds.). Springer, 626–643.

[3] Vitalik Buterin, Ian Goldberg, Aniket Kate, and Ivan Pryvalov. 2023. Relicwrapper library. Retrieved September 15, 2023 from https://gitlab.com/pryvalov/relicwrapper

[4] Jan Camenisch and Markus Stadler. 1997. Proof systems for general statements about discrete logarithms. *Technical Report/ETH Zurich, Department of Computer Science* 260 (1997).

[5] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. 2016. BeleniosRF: A Non-interactive Receipt-Free Electronic Voting Scheme. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. 1614–1625.

[6] David Chaum and Torben P. Pedersen. 1992. Wallet Databases with Observers. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings (Lecture Notes in Computer Science, Vol. 740)*, Ernest F. Brickell (Ed.). Springer, 89–105. https://doi.org/10.1007/3-540-48071-4_7

[7] Benoît Chevallier-Mames, Pierre-Alain Fouque, David Pointcheval, Julien Stern, and Jacques Traoré. 2010. On Some Incompatible Properties of Voting Schemes. In *Towards Trustworthy Elections, New Directions in Electronic Voting (Lecture Notes in Computer Science, Vol. 6000)*, David Chaum, Markus Jakobsson, Ronald L. Rivest, Peter Y. A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida (Eds.). Springer, 191–199.

[8] Sherman S. M. Chow, Joseph K. Liu, and Duncan S. Wong. 2008. Robust Receipt-Free Election System with Ballot Secrecy and Verifiability. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2008, San Diego, California, USA, 10th February - 13th February 2008*. The Internet Society. https://www.ndss-symposium.org/ndss2008/robust-receipt-free-election-system-ballot-secrecy-and-verifiability/

[9] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. 2008. Civitas: Toward a Secure Voting System. In *2008 IEEE Symposium on Security and Privacy (S&P 2008), 18-21 May 2008, Oakland, California, USA*. IEEE Computer Society, 354–368. https://doi.org/10.1109/SP.2008.32

[10] Véronique Cortier, Alicia Filipiak, and Joseph Lallemand. 2019. BeleniosVS: Secrecy and Verifiability Against a Corrupted Voting Device. In *32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019*. IEEE, 367–381. https://doi.org/10.1109/CSF.2019.00032

[11] Véronique Cortier, David Galindo, Ralf Küsters, Johannes Müller, and Tomasz Truderung. 2016. SoK: Verifiability Notions for E-Voting Protocols. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*. 779–798.

[12] Véronique Cortier, Pierrick Gaudry, and Stéphane Glondu. 2019. Belenios: A Simple Private and Verifiable Electronic Voting System. In *Foundations of Security, Protocols, and Equational Reasoning - Essays Dedicated to Catherine A. Meadows (Lecture Notes in Computer Science, Vol. 11565)*, Joshua D. Guttman, Carl E. Landwehr, José Meseguer, and Dusko Pavlovic (Eds.). Springer, 214–238. https://doi.org/10.1007/978-3-030-19052-1_14

[13] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. 2001. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology—CRYPTO'94: 14th Annual International Cryptology Conference Santa Barbara, California, USA August 21–25, 1994 Proceedings*. Springer, 174–187.

[14] Chris Culnane and Steve A. Schneider. 2014. A Peered Bulletin Board for Robust Use in Verifiable Voting Systems. In *IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014*. IEEE Computer Society, 169–183. https://doi.org/10.1109/CSF.2014.20

[15] Edouard Cuvelier, Olivier Pereira, and Thomas Peters. 2013. Election Verifiability or Ballot Privacy: Do We Need to Choose?. In *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings (Lecture Notes in Computer Science, Vol. 8134)*, Jason Crampton, Sushil Jajodia, and Keith Mayes (Eds.). Springer, 481–498. https://doi.org/10.1007/978-3-642-40203-6_27

[16] Damien Desfontaines and Balázs Pejó. 2020. SoK: Differential privacies. *Proc. Priv. Enhancing Technol.* 2020, 2 (2020), 288–313. https://doi.org/10.2478/popets-2020-0028

[17] Henri Devillez, Olivier Pereira, and Thomas Peters. 2022. How to Verifiably Encrypt Many Bits for an Election?. In *Computer Security–ESORICS 2022: 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26–30, 2022, Proceedings, Part II*. Springer, 653–671.

[18] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 3571–3580. https://proceedings.neurips.cc/paper/2017/hash/253614bbac999b38b5b60cae531c4969-Abstract.html

[19] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 4052)*, Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener (Eds.). Springer, 1–12. https://doi.org/10.1007/11787006_1

[20] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, Gail-Joon Ahn, Moti Yung, and Ninghui Li (Eds.). ACM, 1054–1067. https://doi.org/10.1145/2660267.2660348

[21] Ehsan Estaji, Thomas Haines, Kristian Gjøsteen, Peter B. Rønne, Peter Y. A. Ryan, and Najmeh Soroush. 2020. Revisiting Practical and Usable Coercion-Resistant Remote E-Voting. In *Electronic Voting - 5th International Joint Conference, E-Vote-ID 2020, Bregenz, Austria, October 6-9, 2020, Proceedings (Lecture Notes in Computer Science, Vol. 12455)*, Robert Krimmer, Melanie Volkamer, Bernhard Beckert, Ralf Küsters, Oksana Kulyk, David Duenas-Cid, and Mikhel Solvak (Eds.). Springer, 50–66. https://doi.org/10.1007/978-3-030-60347-2_4

[22] Federal Chancellery FCh. 2022. E-voting: Results of the first independent examination available. https://www.bk.admin.ch/bk/en/home/dokumentation/medienmitteilungen.msg-id-88085.html (accessed 23.01.2023).

[23] Taher El Gamal. 1984. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *Advances in Cryptology, Proceedings of CRYPTO '84*. 10–18.

[24] Simson L. Garfinkel, John M. Abowd, and Sarah Powazek. 2018. Issues Encountered Deploying Differential Privacy. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society, WPES@CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, David Lie, Mohammad Mannan, and Aaron Johnson (Eds.). ACM, 133–137. https://doi.org/10.1145/3267323.3268949

[25] Rosario Gennaro. 1995. Achieving Independence Efficiently and Securely. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing, Ottawa, Ontario, Canada, August 20-23, 1995*, James H. Anderson (Ed.). ACM, 130–136. https://doi.org/10.1145/224964.224979

[26] Kristian Gjøsteen, Thomas Haines, and Morten Rotvold Solberg. 2021. Efficient mixing of arbitrary ballots with everlasting privacy: How to verifiably mix the PPATC scheme. In *Nordic Conference on Secure IT Systems*. Springer, 92–107.

[27] Thomas Haines, Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. 2020. How Not to Prove Your Election Outcome. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 644–660. https://doi.org/10.1109/SP40000.2020.00048

[28] Thomas Haines, Rafieh Mosaheb, Johannes Müller, and Ivan Pryvalov. 2023. SoK: Secure E-Voting with Everlasting Privacy. *Proc. Priv. Enhancing Technol.* 2023 (2023).

[29] Thomas Haines and Johannes Müller. 2020. SoK: Techniques for Verifiable Mix Nets. In *33rd IEEE Computer Security Foundations Symposium, CSF 2020, Boston, MA, USA, June 22-26, 2020*. IEEE, 49–64. https://doi.org/10.1109/CSF49147.2020.00012

[30] Thomas Haines and Johannes Müller. 2021. A Novel Proof of Shuffle: Exponentially Secure Cut-and-Choose. In *Information Security and Privacy - 26th Australasian Conference, ACISP 2021, Proceedings (Lecture Notes in Computer Science)*.

[31] Thomas Haines, Johannes Müller, and Iñigo Querejeta-Azurmendi. 2023. Scalable Coercion-Resistant E-Voting under Weaker Trust Assumptions. In *SAC '23: The 38th ACM/SIGAPP Symposium on Applied Computing, 2023*. ACM.

[32] Sven Heiberg, Tarvi Martens, Priit Vinkel, and Jan Willemson. 2016. Improving the Verifiability of the Estonian Internet Voting Scheme. In *Electronic Voting - First International Joint Conference, E-Vote-ID 2016, Bregenz, Austria, October 18-21, 2016, Proceedings (Lecture Notes in Computer Science, Vol. 10141)*, Robert Krimmer, Melanie Volkamer, Jordi Barrat, Josh Benaloh, Nicole J. Goodman, Peter Y. A. Ryan, and Vanessa Teague (Eds.). Springer, 92–107. https://doi.org/10.1007/978-3-319-52240-1_6

[33] Lucca Hirschi, Lara Schmid, and David A. Basin. 2021. Fixing the Achilles Heel of E-Voting: The Bulletin Board. In *34th IEEE Computer Security Foundations Symposium, CSF 2021, Dubrovnik, Croatia, June 21-25, 2021*. IEEE, 1–17. https://doi.org/10.1109/CSF51468.2021.00016

[34] Ari Juels, Dario Catalano, and Markus Jakobsson. 2005. Coercion-resistant electronic elections. In *Proceedings of the 2005 ACM Workshop on Privacy in*

the Electronic Society, WPES 2005, Alexandria, VA, USA, November 7, 2005, Vijay Atluri, Sabrina De Capitani di Vimercati, and Roger Dingledine (Eds.). ACM, 61–70. https://doi.org/10.1145/1102199.1102213

[35] Aggelos Kiayias, Annabell Kuldmaa, Helger Lipmaa, Janno Siim, and Thomas Zacharias. 2018. On the Security Properties of e-Voting Bulletin Boards. In *Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings (Lecture Notes in Computer Science, Vol. 11035)*, Dario Catalano and Roberto De Prisco (Eds.). Springer, 505–523. https://doi.org/10.1007/978-3-319-98113-0_27

[36] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. 2015. End-to-End Verifiable Elections in the Standard Model. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 9057)*, Elisabeth Oswald and Marc Fischlin (Eds.). Springer, 468–498. https://doi.org/10.1007/978-3-662-46803-6_16

[37] Ios Kotsogiannis, Stelios Doudalis, Samuel Haney, Ashwin Machanavajjhala, and Sharad Mehrotra. 2020. One-sided Differential Privacy. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020.* IEEE, 493–504. https://doi.org/10.1109/ICDE48307.2020.00049

[38] Oksana Kulyk, Vanessa Teague, and Melanie Volkamer. 2015. Extending Helios Towards Private Eligibility Verifiability. In *E-Voting and Identity - 5th International Conference, VoteID 2015, Bern, Switzerland, September 2-4, 2015, Proceedings (Lecture Notes in Computer Science, Vol. 9269)*, Rolf Haenni, Reto E. Koenig, and Douglas Wikström (Eds.). Springer, 57–73. https://doi.org/10.1007/978-3-319-22270-7_4

[39] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2010. Accountability: Definition and Relationship to Verifiability. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010.* 526–535.

[40] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2011. Verifiability, Privacy, and Coercion-Resistance: New Insights from a Case Study. In *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA.* 538–553.

[41] Wouter Lueks, Iñigo Querejeta-Azurmendi, and Carmela Troncoso. 2020. VoteAgain: A Scalable Coercion-Resistant Voting System. In *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, Srdjan Capkun and Franziska Roesner (Eds.). USENIX Association, 1553–1570. https://www.usenix.org/conference/usenixsecurity20/presentation/lueks

[42] David Mestel, Johannes Müller, and Pascal Reisert. 2022. How Efficient are Replay Attacks against Vote Privacy? A Formal Quantitative Analysis. In *35th IEEE Computer Security Foundations Symposium, CSF 2022, Haifa, Israel, August 7-10, 2022.* IEEE, 179–194.

[43] Johannes Müller. 2022. Breaking and Fixing Vote Privacy of the Estonian E-Voting Protocol. In *Financial Cryptography and Data Security - FC 2022 International Workshops.* https://orbilu.uni.lu/bitstream/10993/49442/1/main.pdf

[44] Johannes Müller and Tomasz Truderung. 2023. A Protocol for Cast-as-Intended Verifiability with a Second Device. *CoRR* abs/2304.09456 (2023). https://doi.org/10.48550/arXiv.2304.09456 arXiv:2304.09456

[45] Office of the United Nations High Commissioner for Human Rights. 2021. Human Rights and Elections: A Handbook on International Human Rights Standards on Elections. (2021). https://www.un-ilibrary.org/content/books/9789214030522

[46] Olivier Pereira. 2022. Individual Verifiability and Revoting in the Estonian Internet Voting System. In *Financial Cryptography and Data Security - FC 2022 International Workshops.* https://eprint.iacr.org/2021/1098

[47] Peter B. Rønne, Peter Y. A. Ryan, and Ben Smyth. 2021. Cast-as-Intended: A Formal Definition and Case Studies. In *Financial Cryptography and Data Security. FC 2021 International Workshops - CoDecFin, DeFi, VOTING, and WTSC, Virtual Event, March 5, 2021, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 12676)*, Matthew Bernhard, Andrea Bracciali, Lewis Gudgeon, Thomas Haines, Ariah Klages-Mundt, Shin'ichiro Matsuo, Daniel Perez, Massimiliano Sala, and Sam Werner (Eds.). Springer, 251–262. https://doi.org/10.1007/978-3-662-63958-0_22

[48] Peter Y. A. Ryan, Peter B. Rønne, and Vincenzo Iovino. 2016. Selene: Voting with Transparent Verifiability and Coercion-Mitigation. In *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 9604)*, Jeremy Clark, Sarah Meiklejohn, Peter Y. A. Ryan, Dan S. Wallach, Michael Brenner, and Kurt Rohloff (Eds.). Springer, 176–192. https://doi.org/10.1007/978-3-662-53357-4_12

[49] Claus-Peter Schnorr. 1991. Efficient Signature Generation by Smart Cards. *J. Cryptol.* 4, 3 (1991), 161–174. https://doi.org/10.1007/BF00196725

[50] István András Seres, Balázs Pejó, and Péter Burcsi. 2022. The Effect of False Positives: Why Fuzzy Message Detection Leads to Fuzzy Privacy Guarantees?. In *International Conference on Financial Cryptography and Data Security.* Springer, 123–148.

[51] Shun Takagi, Yang Cao, and Masatoshi Yoshikawa. 2021. Asymmetric Differential Privacy. *CoRR* abs/2103.00996 (2021). arXiv:2103.00996 https://arxiv.org/abs/2103.00996

[52] Björn Terelius and Douglas Wikström. 2010. Proofs of Restricted Shuffles. In *Progress in Cryptology - AFRICACRYPT 2010, Third International Conference on Cryptology in Africa (Lecture Notes in Computer Science, Vol. 6055)*, Daniel J. Bernstein and Tanja Lange (Eds.). Springer, 100–113.

A COMPUTATIONAL MODEL

We formally model DeVoS in a general computational framework that we can use both to analyze both verifiability and privacy properties.

Background. Our underlying computational model (see, e.g., [11] for details) introduces the notion of a process which can be used to model protocols. Essentially, a process $\hat{\pi}_{\mathsf{P}}$ modeling some protocol P is a set of interacting ppt Turing machines which capture the honest behavior of protocol participants. The protocol P runs alongside an adversary A, modeled via another process π_{A}, which controls the network and may corrupt protocol participants; here we assume static corruption. We write $\pi{=}(\hat{\pi}_{\mathsf{P}} \| \pi_{\mathsf{A}})$ for the combined process.

Modeling of DeVoS. The DeVoS voting protocol can be modeled in a straightforward way as a protocol $\mathsf{P}_{\mathsf{DeVoS}}(n, C, \mu)$ in the above sense, as detailed next. By n we denote the number of voters V_i. By μ we denote a probability distribution on the set of possible choices C. An honest voter makes her choice according to this distribution. This choice is called the *actual choice* of the voter.

In our model of DeVoS, the voting authority EA is part of an additional agent, the scheduler S. Besides playing the role of the authority, S schedules all other agents in a run according to the protocol phases. We assume that S and the public bulletin board PBB are honest, i.e., they are never corrupted. While S is only a virtual entity, in reality, PBB should be implemented in a distributed way (see, e.g., [14, 33, 35]).

B VERIFIABILITY

We analyze verifiability of DeVoS. For this purpose, we use the generic verifiability definition proposed in [39]. We recall this definition in what follows. Afterwards, we precisely state under which assumptions DeVoS is verifiable according to [39].

Framework. The verifiability definition [39] is centered around a "virtual" entity, called the *judge* J. In reality, the program of the judge can be run by any party, including external observers and even voters themselves. In a given protocol run, the judge J takes as input solely public information (e.g., the zero-knowledge proofs in DeVoS published on the bulletin board) and then performs certain checks. If all checks pass, the judge accepts the protocol run, and rejects it otherwise.

In the context of e-voting, for verifiability to hold, the judge should only accept a run if "the announced election result corresponds to the actual choices of the voters". This statement is formalized by the notion of a *goal* γ of a protocol P. A goal γ is simply a set of protocol runs for which the above statement is true, where the description of a run includes the description of the protocol, the adversary with which the protocol is run, and the random coins used by these entities.

According to [39], a goal γ is *verifiable* by the judge J in a protocol P if and only if J accepts a run r of P in which the goal γ is violated (i.e., $r \notin \gamma$) with at most negligible probability (in the security parameter). In order to capture this notion formally, we denote by $\Pr[(\hat{\pi}_{\mathsf{P}}\|\pi_{\mathsf{A}})^{(\ell)} \mapsto \neg\gamma, (\mathsf{J}\colon \mathsf{accept})]$ the probability that a run of the protocol along with an adversary π_{A} (and a security parameter ℓ) will produce a run which is not in γ, but in which J (still) returns accept. This probability should be negligible.

Definition B.1 (Verifiability [39]). We say that *a goal γ is verifiable by the judge* J *in a protocol* P if for all adversaries π_{A}, the probability

$$\Pr[(\hat{\pi}_{\mathsf{P}}\|\pi_{\mathsf{A}})^{(\ell)} \mapsto \neg\gamma, (\mathsf{J}\colon \mathsf{accept})]$$

is negligible as a function of ℓ.

For our subsequent verifiability analysis of DeVoS, we instantiate the verifiability definition with the goal $\gamma(\varphi)$ proposed in [11]. This goal captures the intuition of γ given earlier. The parameter φ is a Boolean formula describing which protocol participants are assumed honest. The goal $\gamma(\varphi)$ is defined formally as described next.

Definition B.2 (Goal $\gamma(\varphi)$ [11]). Let P be a voting protocol. Let I_h and I_d denote the set of honest and dishonest voters, respectively, in a given protocol run. Then, $\gamma(\varphi)$ consists of all those runs of the voting protocol P where either

- φ is false (e.g., the adversary corrupted a voter that is assumed to be honest), or
- φ holds true and there exist (valid) dishonest choices $(v_i)_{i\in I_d}$ such that the election result equals $(v_i)_{i\in I_h \cup I_d}$ (modulo permutation), where $(v_i)_{i\in I_h}$ are the honest voters' choices.

Result. We prove verifiability of DeVoS under the following assumptions:

(V1) The PKE scheme $\mathcal{E}$ with re-randomisation is correct (for verifiability, IND-CPA-security is not needed).
(V2) The function $f\colon \{0,1\}^* \rightarrow \{0,1\}^*$ is one-way.
(V3) $\pi_{\mathsf{KeyGen}} and \pi_{\mathsf{T}}$ are NIP systems (for verifiability, ZK is not needed) and π_{Enc} is a NIZKP.
(V4) The voting authority EA, the public bulletin board PBB, and the judge J are honest:

$$\varphi = \mathsf{hon}(\mathsf{S}) \wedge \mathsf{hon}(\mathsf{PBB}) \wedge \mathsf{hon}(\mathsf{J}).$$

(V5) The messages from EA to any V_i remain private.

The verification procedure J of DeVoS essentially involves checking the NIZKPs published on the bulletin board PBB and whether a voter complained that her submitted ballot was not appended: if one of these checks fails, the protocol run and hence the result are rejected. Now, the following theorem states that the probability that in a run of DeVoS an honest voter's vote has been dropped or manipulated if φ holds true (i.e., $\gamma(\varphi)$ is broken) but the protocol run is nevertheless accepted by J is negligible.

Theorem B.3 (Verifiability). *Under the assumptions (V1) to (V5) stated above, the goal $\gamma(\varphi)$ is verifiable in the protocol $\mathsf{P}_{DeVoS}(n, C, \mu)$ by the judge* J.

Proof. Assume that assumptions (V1) to (V5) hold true. In order to prove Theorem B.3, we need to show the following implication. If the judge J outputs accept in a given protocol run of DeVoS (in which (V1) to (V4) are satisfied), then there exist (valid) dishonest choices $(v_i)_{i\in I_d}$ such that the election result equals $(v_{\sigma(i)})_{i\in I_h \cup I_d}$, where $(v_i)_{i\in I_h}$ are the honest voters' choices and σ is some permutation.

Assume that we are in a run r of DeVoS in which the J outputs accept. Then, due to the specification of J, each NIZKP published on B is valid.

Let V_i be an arbitrary *honest* voter who chose v_i and thus submitted ballot $(\mathsf{e}', \pi_{\mathsf{Enc}})$ where $\mathsf{e}' \in \mathsf{Enc}(\mathsf{pk}, v_i)$. Let e'' be the ciphertext appended by the posting trustee PT to V_i's vector $\vec{e}_i$ right after e' (if any). Since each NIZKP is valid, due to the soundness of π_{Enc} (assumption (V3)), it follows that

(1) $\mathsf{e}'' \in \mathsf{Enc}(\mathsf{pk}, v^*)$ for some v^*, or
(2) $\mathsf{e}'' \in \mathsf{ReRand}(\mathsf{pk}, \mathsf{e}')$

holds true. In case (1), due to the knowledge soundness property of π_{Enc} and the fact that PT posted a valid proof for e'', it follows that PT knows a witness (ssk_i, v^*, r) for the relation $\mathsf{vk}_i = f(\mathsf{ssk}_i) \wedge \mathsf{e}'' = \mathsf{Enc}(\mathsf{pk}, v^*; r) \wedge v^* \in C$. However, since V_i is honest, there exist two possibilities for PT to learn ssk_i: extracting ssk_i from $\mathsf{vk}_i = f(\mathsf{ssk}_i)$ or extracting ssk_i from V_i's NIZKP π_{Enc} for e'. Due to the one-way property of f (assumption (V2)), the ZK property of π_{Enc} (assumption (V3)), and the fact that the channel from EA to V_i, over which the credentials ssk_i are sent, is private (assumption (V5)), we can deduce that case (1) can occur in at most a negligible set of protocol runs of DeVoS. Hence, with overwhelming probability in the security parameter ℓ, we have that case (2) occurs.

By the re-randomisation property of $\mathcal{E}$ (assumption (V1)), it therefore follows via induction that the last ciphertext e_i in $\vec{e}_i$, which is V_i's input to the subsequent tallying phase, encrypts V_i's choice, i.e., $\mathsf{e}_i \in \mathsf{Enc}(\mathsf{pk}, v_i)$.

Now, let V_i be an arbitrary *dishonest* voter and let e_i be the last ciphertext in $\vec{e}_i$. Due to the soundness of π_{Enc} (assumption (V3)), there exist two possible cases:

(1) e_i is a re-randomisation of the previous ciphertext, or
(2) e_i is a fresh encryption of some $v_i \in C$.

In case (1), we can again distinguish between the same cases for the previous ciphertext, and so on. Due to the re-randomisation property of $\mathcal{E}$ (assumption (V1)), it therefore follows that $\mathsf{e}_i \in \mathsf{Enc}(\mathsf{pk}, v_i)$ for some $v_i \in C$.

From what we have shown above, we can deduce that (with overwhelming probability) the input to the tallying phase consists of ciphertexts $(\mathsf{e}_i)_{i\in I_h \cup I_d}$, where for each $i \in I_h$ the respective ciphertext e_i encrypts V_i's intended choice v_i, and where for each $i \in I_d$ the respective ciphertext e_i encrypts some v_i.

Since the judge accepted the protocol run, the tallier's NIZKP π_{T} to prove the correctness of the tallying phase is valid. Due to the soundness of π_{T} (assumption (V3)), Theorem B.3 follows.

C VOTE PRIVACY

We analyze vote privacy of DeVoS. We show that the privacy level of DeVoS is ideal under minimal assumptions. To this end, we use the privacy definition for e-voting protocols proposed in [40]. In what follows, we first recall this definition, and then state the privacy result of DeVoS for the shuffling-based version (because we showed that this mode is practically advantageous over the homomorphic mode, see Section 7).

Framework. The definition proposed in [40] formalizes privacy of an e-voting protocol as the inability of an adversary to distinguish whether some voter $\mathsf{V}_{\mathsf{obs}}$ (the *voter under observation*), who runs her honest program, voted for choice v_0 or choice v_1.

To define this notion formally, we first introduce the following notation for an arbitrary e-voting protocol P. Given a voter $\mathsf{V}_{\mathsf{obs}}$ and $v \in C$, we consider instances of P of the form $(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)\|\pi^*\|\pi_{\mathsf{A}})$ where $\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)$ is the honest program of the voter $\mathsf{V}_{\mathsf{obs}}$ under observation who takes v as her choice, π^* is the composition of programs of the remaining parties in P, and π_{A} is the program of the adversary. In the case of DeVoS, π^* includes the scheduler S, the public bulletin board PBB, a set of uncorrupted voters, the mix server M, and the decryption trustee T.

Let $\Pr[(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)\|\pi^*\|\pi_{\mathsf{A}})^{(\ell)} \mapsto 1]$ denote the probability that the adversary writes the output 1 on some dedicated tape in a run of $(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)\|\pi^*\|\pi_{\mathsf{A}})$ with security parameter ℓ and some $v \in C$, where the probability is taken over the random coins used by the parties in $(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)\|\pi^*\|\pi_{\mathsf{A}})$. Now, vote privacy is defined as follows.

Definition C.1 (Privacy). Let P be a voting protocol, $\mathsf{V}_{\mathsf{obs}}$ be the voter under observation, and $\delta \in [0, 1]$. Then, P *achieves δ-privacy*, if for all choices $v_0, v_1 \in C$ and all adversaries π_{A} the difference

$$\Pr[(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v_0)\|\pi^*\|\pi_{\mathsf{A}})^{(\ell)} \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v_1)\|\pi^*\|\pi_{\mathsf{A}})^{(\ell)} \mapsto 1]$$

is δ-bounded[11] as a function of the security parameter 1^ℓ.

In other words, the level δ is an upper bound of an arbitrary adversary's advantage to "break" vote privacy. Therefore, δ should be as small as possible. Note, however, that even for an ideal e-voting protocol with a completely passive adversary, δ might not be 0: for example, there might be a non-negligible chance that all honest voters, including the voter under observation, voted for the same candidate, in which case the adversary can easily derive from the final election result how the voter under observation voted.

Result. We now state that DeVoS provides ideal vote privacy, essentially under the assumption that the mix server M and the decryption trustee T are honest.

More specifically, the formal privacy result for DeVoS is formulated w.r.t. an ideal voting protocol $\mathcal{I}_{\mathsf{voting}}(n, C, \mu, v)$ (see Fig. 4). In this protocol, all n voters pick their candidates according to the distribution μ, and the voter under observation votes for v. The ideal protocol outputs these choices permuted uniformly at random. The privacy level $\delta^{\mathsf{ideal}}_{(n,C,\mu,v)}$ this ideal protocol has depending on the given parameters was derived in [40].

To prove that the privacy level of DeVoS is ideal, we make the following assumptions about the primitives we use and the protocol parties involved (see Section 4):

(VP1) The PKE scheme with re-randomization is IND-CPA-secure.
(VP2) The function $f\colon \{0,1\}^* \to \{0,1\}^*$ is one-way.
(VP3) $\pi_{\mathsf{KeyGen}}, \pi_{\mathsf{Shuffle}}, \pi_{\mathsf{Dec}}$ are NIZKPs and π_{Enc} is a NIZKPoK.
(VP4) The scheduler S, the public bulletin board PBB, the mix server M, the decryption trustee T, and at least n_h voters are honest.
(VP5) The messages from EA to any V_i remain private.

$\underline{\mathcal{I}_{\mathsf{voting}}(n, C, \mu, v_b)}$

Parameters:

- Number of voters n
- Choice space C
- Probability distribution μ over C

On init from S do:

(1) $\forall i \in \{1, \dots, n\}$: store $v_i \overset{\mu}{\longleftarrow} C$.
(2) Store $v_{n+1} \longleftarrow v_b$
(3) Return success.

On $(\mathsf{compute}, b)$ from S do:

(1) If $b = 0$, return $\perp$.
(2) Choose permutation $\sigma \overset{r}{\longleftarrow} [n+1]^{[n+1]}$.
(3) Return res $\longleftarrow (v_{\sigma(i)})_{i \in [n+1]}$.

Figure 4: Ideal privacy functionality for voting protocol.

Now, the following privacy theorem says that the privacy level of DeVoS is ideal under the previous assumptions.

Theorem C.2 (Privacy). *Let C be the set of valid choices, excluding abstention. Then, under the assumptions (VP1) to (VP5) stated above, the voting protocol $\mathsf{P}_{DeVoS}(n, C, \mu)$ achieves a privacy level of $\delta^{\mathsf{ideal}}_{(n_h,C,\mu)}$.*

Proof. In this section, we prove Theorem C.2 which establishes the privacy level of DeVoS. This privacy level can be expressed using the privacy level $\delta^{\mathsf{ideal}}_{(n,C,\mu)}$ of the voting protocol $\mathcal{I}_{\mathsf{voting}}(n, C, \mu)$ with ideal privacy (see Fig. 4).

Overview of the proof. Recall that, in order to prove the theorem for the protocol DeVoS with n voters, choice space C, voting distribution μ, and voter under observation $\mathsf{V}_{\mathsf{obs}}$, we have to show that

$$\left|\Pr[(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v_0)\|\pi^*) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v_1)\|\pi^*) \mapsto 1]\right|$$

is $\delta^{\mathsf{ideal}}_{(n_h,C,\mu)}$-bounded as a function of the security parameter ℓ, for all $v_0, v_1 \in C$, all programs π^* of the remaining parties such that the scheduler S, the public bulletin board PBB, the mix server M, the decryption trustee T, and at least n_h voters are honest in π^* (excluding the voter under observation $\mathsf{V}_{\mathsf{obs}}$).

We can split up the composition π^* in its honest and its (potentially) dishonest part. Let HV be the set of all honest voters (without the voter under observation) and $\hat{\pi}_{\mathsf{HV}}$ be the composition of their honest programs. Therefore, the honest part, which we denote by

$$\hat{\pi}_{\mathsf{H}} = \hat{\pi}_{\mathsf{S}}\|\hat{\pi}_{\mathsf{PBB}}\|\hat{\pi}_{\mathsf{M}}\|\hat{\pi}_{\mathsf{T}}\|\hat{\pi}_{\mathsf{HV}},$$

consists of the honest programs $\hat{\pi}_{\mathsf{S}}, \hat{\pi}_{\mathsf{B}}, \hat{\pi}_{\mathsf{M}}, \hat{\pi}_{\mathsf{T}}, \hat{\pi}_{\mathsf{HV}}$ of the scheduler S, the public bulletin board PBB, the mix server M, decryption trustee T, and the honest voters HV, respectively. By $\hat{\pi}_{\mathsf{H}}(v)$ we will denote the composition of all honest programs including the program of the voter under observation $\mathsf{V}_{\mathsf{obs}}$, i.e., $\hat{\pi}_{\mathsf{H}}(v) = \hat{\pi}_{\mathsf{H}}\|\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)$. All remaining parties are subsumed by the adversarial process π_{A}. This means that we can write $\hat{\pi}_{\mathsf{V}_{\mathsf{obs}}}(v)\|\pi^*$ as $\hat{\pi}_{\mathsf{H}}(v)\|\pi_{\mathsf{A}}$.

In order to prove the result, we use a sequence of games. We fix $v \in C$ and start with Game 0 which is simply the process $\hat{\pi}_{\mathsf{H}}(v)\|\pi_A$.

[11] A function f is *δ-bounded* if, for every $c > 0$, there exists ℓ_0 such that $|f(\ell)| \leq \delta + \ell^{-c}$ for all $\ell > \ell_0$.

Step by step, we transform Game 0 into Game 6 which is the composition $\hat{\pi}_{\mathsf{H}}^{6}(v) \| \pi_A$ for some process $\hat{\pi}_{\mathsf{H}}^{6}(v)$ and the same adversarial process π_A. Game 6 will be proven indistinguishable from Game 0 from the adversary's point of view, which means that

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{0}(v) \| \pi_A) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v) \| \pi_A) \mapsto 1]\right|$$

is negligible for a fixed $v \in C$ (as a function of the security parameter). On the other hand, it will be straightforward to show that in Game 6 for arbitrary $v_0, v_1 \in C$, the distance

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v_0) \| \pi_A) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v_1) \| \pi_A) \mapsto 1]\right|$$

is bounded by $\delta^{\mathsf{ideal}}_{(n_h, C, \mu)}$ because $\hat{\pi}_{\mathsf{H}}^{6}(v_0)$ and $\hat{\pi}_{\mathsf{H}}^{6}(v_1)$ use the ideal voting protocol for n_h honest voters. Using the triangle inequality, we can therefore deduce that

$$|\Pr[(\hat{\pi}_{\mathsf{H}}(v_0) \| \pi_A) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}(v_1) \| \pi_A) \mapsto 1]|$$

is $\delta^{\mathsf{ideal}}_{(n_h, C, \mu)}$-bounded for all $v_0, v_1 \in C$ (as a function of the security parameter).

In each of the games described below, we merely specify the differences to the respective previous game. Apart from these modifications, the respective games are the same.

Game 0. In what follows, we write $\hat{\pi}_{\mathsf{H}}^{0}(v)$ for $\hat{\pi}_{\mathsf{H}}(v)$ and consider $\hat{\pi}_{\mathsf{H}}^{0}(v)$ as one atomic process (one program) and not as a composition of processes.[12] Now, Game 0 is the process $\hat{\pi}_{\mathsf{H}}^{0}(v) \| \pi_A$. △

In the first four games, we will simulate the NIZKPs of all honest parties.

Game 1. In the setup phase, the decryption trustee T (as a subprocess of $\hat{\pi}_{\mathsf{H}}^{1}(v)$) exploits the ZK property to simulate the NIZKP of correct key generation π_{KeyGen}. △

Game 2. In the tallying phase, the mix server M (as a subprocess of $\hat{\pi}_{\mathsf{H}}^{2}(v)$) exploits the ZK property to simulate the NIZKP of correct shuffling π_{Shuffle}. △

Game 3. In the tallying phase, the decryption trustee T (as a subprocess of $\hat{\pi}_{\mathsf{H}}^{3}(v)$) exploits the ZK property to simulate the NIZKP of correct decryption π_{Dec}. △

Game 4. In the submission phase, each honest voter V_i and the voter under observation $\mathsf{V}_{\mathsf{obs}}$ (as subprocesses of $\hat{\pi}_{\mathsf{H}}^{4}(v)$) exploit the ZK property to simulate the NIZKP of correct encryption π_{Enc}. △

In the next four game, we add 'stop events'.

Game 5. If $(\mathsf{pk}, \mathsf{sk}) \notin \mathsf{KeyGen}$, where pk is the public key posted by T on PBB, then the simulator aborts.

Game 6. If for one of the honest voters V_i (including the voter under observation $\mathsf{V}_{\mathsf{obs}}$), V_i's input ciphertext e_i to the tallying phase is not an encryption to the voter's choice v_i, i.e., $e_i \notin \mathsf{Enc}(\mathsf{pk}, v_i)$, then the simulator aborts. △

Game 7. Let $(e_i)_{i=1}^{n}$ be the input to the tallying phase, and let $(e'_i)_{i=1}^{n}$ be the output of M. If there does not exist a permutation σ over $[n]$ such that $e'_{\sigma(i)} \in \mathsf{ReRand}(e_i)$ for all $i \in [n]$, then the simulator aborts.

Game 8. Let $(e'_i)_{i=1}^{n}$ be the output of M, and let $(v'_i)_{i=1}^{n}$ be the output of T. If there exists $i \in [n]$ such that $e'_i \notin \mathsf{Enc}(\mathsf{pk}, v'_i)$, then the simulator aborts.

In the next game, we invoke the ideal voting functionality to calculate the result for the honest voters.

Game 9. At the beginning of the submission phase, the simulator runs the ideal voting functionality $\mathcal{I}_{\mathsf{voting}}(n_h, C, \mu, v)$ to compute the honest voters' (including the observed voter $\mathsf{V}_{\mathsf{obs}}$) permuted choices. The voter under observation then runs her voting procedure with the $(n+1)$-th output of $\mathcal{I}_{\mathsf{voting}}$, and the remaining honest voters run their voting procedure with the the associated first n outputs of $\mathcal{I}_{\mathsf{voting}}$. △

Lemma C.3. *Let $i \in \{0, 1, 2, 3\}$. Then Game i and Game $i+1$ are computationally indistinguishable, i.e., we have that*

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{i}(v) \| \pi_{\mathsf{A}}) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{i+1}(v) \| \pi_{\mathsf{A}}) \mapsto 1]\right|$$

is negligible (as a function of the security parameter).

Proof. This fact follows from the ZK property of the NIZKPs (VP3) and the assumption that M and T are honest (VP4).

Lemma C.4. *Let $i \in \{4, 6, 7\}$. Then Game i and Game $i+1$ are computationally indistinguishable, i.e., we have that*

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{i}(v) \| \pi_{\mathsf{A}}) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{i+1}(v) \| \pi_{\mathsf{A}}) \mapsto 1]\right|$$

is negligible (as a function of the security parameter).

Proof. This fact follows from the honesty of M and T (VP4).

Lemma C.5. *Game 5 and Game 6 are computationally indistinguishable, i.e., we have that*

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{4}(v) \| \pi_{\mathsf{A}}) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{5}(v) \| \pi_{\mathsf{A}}) \mapsto 1]\right|$$

is negligible.

Proof. This fact follows from the soundness of π_{Enc} (VP3), the one-way property of f (VP2), the honesty of EA (VP4), and the assumption that the channel from EA to each honest V_i is untappable (VP5).

Lemma C.6. *Game 8 and Game 9 are computationally indistinguishable, i.e., we have that*

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{5}(v) \| \pi_{\mathsf{A}}) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v) \| \pi_{\mathsf{A}}) \mapsto 1]\right|$$

is negligible.

Proof. From Games 1–4, it follows that all NIZKPs by honest parties are simulated. From Games 5–8, it follows that the individual choice of each honest V_i, including the voter under observation, is included in the final result. Note that since π_{Enc} is a proof of knowledge (VP3) and Enc is CPA-secure, it follows that the honest voters' ciphertexts are non-malleable (see, e.g., [2]). Hence, if the adversary's advantage to distinguish between Game 8 and Game 9 was significant, then this would contradict the NM-CPA-security of Enc augmented with π_{Enc}.

Lemma C.7. *For Game 9, we have that*

$$\left|\Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v_0) \| \pi_A) \mapsto 1] - \Pr[(\hat{\pi}_{\mathsf{H}}^{6}(v_1) \| \pi_A) \mapsto 1]\right|$$

is bounded by $\delta^{\mathsf{ideal}}_{(n_h, C, \mu)}$.

[12] This is w.l.o.g. since every (sub-)process can be simulated by a single program.

Proof. This follows from the fact that the simulator uses the ideal voting functionality $\mathcal{I}_{\text{voting}}(n_h, C, \mu, v_b)$ to compute the honest voters' choices.

D STRONG PARTICIPATION PRIVACY AND DENIABLE VOTE UPDATING

We analyze strong participation privacy and deniable vote updating of DeVoS. As in our analysis of vote privacy (App. C), we restrict our attention to the mix-net version of DeVoS.

Frameworks. We use the coercion-resistance definition in [40] which assumes that a coerced voter has a certain goal γ that she would try to achieve in absence of coercion. Formally, γ is a property of the voting protocol P. In the case of strong participation privacy, γ expresses that the coerced voter wants to vote for an arbitrary candidate, so γ contains all runs in which the coerced voter voted for an arbitrary candidate and this vote is in fact counted. In the case of deniable vote updating, γ expresses that the coerced voter wants to vote for a specific candidate, say A.

In this definition of coercion-resistance, the coercer wants the coerced voter to run a certain strategy, the dummy strategy dum, instead of the program an honest voter would run. In the case of strong participation privacy, if the coerced voter runs dum, then she abstains from voting. In the case of deniable vote updating, if the coerced voter runs dum, then she votes for the candidate that the coercer told her to vote for.

Now, for a protocol to be coercion-resistance, this definition requires that there exists a counter-strategy α that the coerced voter can run instead of dum such that (i) the coerced voter achieves her own goal γ, with overwhelming probability, by running α, and (ii) the coercer is not able to distinguish whether the coerced voter runs dum or α. Similarly to the vote privacy definition (see Appendix C), this definitions measures the ability of the coercer to distinguish between these two cases. Hence, α has to simulate dum while at the same time make sure that γ is achieved. In the case of strong participation privacy, α is the prescribed program that an honest voter runs in DeVoS if she votes for an arbitrary candidate. In the case of deniable vote updating, α is the prescribed program that an honest voter runs in DeVoS if she votes for her favorite candidate.

Definition D.1 (Coercion-resistance). Let P be a voting protocol, γ be a property of P, and $\delta \in [0, 1]$. Then, P is *δ-coercion-resistant w.r.t. γ* if there exists α such that for all adversaries π_A:

(1) $\Pr[(\alpha \| \pi^* \| \pi_A)^{(\ell)} \mapsto \gamma]$ is overwhelming as a function of the security parameter 1^ℓ.
(2) The advantage

$$\Pr[(\alpha \| \pi^* \| \pi_A)^{(\ell)} \mapsto 1] - \Pr[(\mathsf{dum} \| \pi^* \| \pi_A)^{(\ell)} \mapsto 1]$$

is δ-bounded as a function of the security parameter 1^ℓ.

Results. We refer to Section 5.4 for the assumptions about the primitives we use and the protocol parties involved (see Section 4).

The following theorem states that the coercion-resistance level of DeVoS for voters who decide to vote (for an arbitrary candidate) is ideal under the previous assumptions.

Theorem D.2 (Strong participation privacy). *Let γ contain all runs in which a voter votes for an arbitrary candidate and this vote is in fact counted. Then, under the assumptions (PP1) to (PP6) stated above, the voting protocol $\mathsf{P}_{DeVoS}(n, C, \mu)$ achieves a coercion-resistance level of $\delta^{\text{ideal}}_{(n_h, C, \mu)}$.*

The following theorem states that the coercion-resistance level of DeVoS for voters who decide to vote for their favorite candidates is ideal under the previous assumptions.

Theorem D.3 (Deniable vote updating). *Let γ contain all runs in which a voter votes for her favorite candidate and this vote is in fact counted. Then, under the assumptions (PP1) to (PP6) stated above, the voting protocol $\mathsf{P}_{DeVoS}(n, C, \mu)$ achieves a coercion-resistance level of $\delta^{\text{ideal}}_{(n_h, C, \mu)}$.*

Proofs (sketches). Both proofs are based on the vote privacy proof presented Appendix C. First, we note that the vote privacy game and the coercion-resistance game coincide in the sense that in both cases, the observer/coercer needs to distinguish between the case that the voter under observation/the coerced voter selects a (valid) choice v_0 or v_1. The only differences is that the coercer unlike the passive observer can give instructions to the coerced voter.

Now, to prove strong participation privacy and deniable vote updating, we extend the sequence of games in the vote privacy proof as follows. We add a new Game 4' between Game 4 and Game 5, which states that the posting trustee PT exploits the ZK property to simulate its NIZKP π_{Enc}. Moreover, we also specify that "abstention" is a choice that the ideal functionality can output. Then, under the additional assumptions that PT is honest (PP4) and that the adversary cannot monitor the channels between honest voters and PT (PP5), it follows that Game 8 and Game 9 are computationally indistinguishable.

It follows that a coercer, who demands a voter to abstain from voting or to vote for a particular candidate, can only deduce information from the final election result about whether the coerced voter obeyed. Since the choice which denotes abstention is the same for all voters, the election result in DeVoS does not reveal more information about the coerced voter's decision (i.e., abstention or not, or vote for the coerced candidate or not) than what can be derived from the output of the ideal functionality. This argument establishes Theorem D.2 and Theorem D.3.

E INSTANTIATION OF DEVOS WITH CONDITIONAL PRIVACY

We describe our ElGamal-based instantiation of DeVoS.

Hardness assumption. The decisional Diffie-Hellman assumption in $\mathbb{G} = \langle g \rangle$ (DDH for short) states the hardness for PPT adversaries of solving the following problem. On input $(g, g^\alpha, g^\beta, Z) \in \mathbb{G}^4$, decide whether $Z = g^{\alpha\beta}$ or a random element in $\mathbb{G}$.

Public-key encryption scheme. We instantiate this central primitive with exponential ElGamal PKE [23].

Definition E.1 (Exponential ElGamal PKE). $\mathsf{KeyGen}(1^\ell)$ chooses cyclic group $\mathbb{G}$ of order p (with $p = \|\ell\|$), generator g of $\mathbb{G}$, then chooses $\alpha \leftarrow \mathbb{Z}_p$ uniformly at random, and returns $\mathsf{pk} = (\mathbb{G}, p, g, h = g^\alpha)$ as public key and $\mathsf{sk} = (\mathbb{G}, p, g, \alpha)$ as secret key. To encrypt a message $m \in \mathbb{Z}_p$ w.r.t. public key $\mathsf{pk} = (\mathbb{G}, p, g, h)$, Enc chooses a random $r \in \mathbb{Z}_p$, and outputs ciphertext $\mathsf{CT} = (u = g^r, v = h^r \cdot g^m) \in$

$w = (\alpha, m, r)$	$w = r$
$\omega_2, a_1, a_2, a_3, z_4 \leftarrow \mathbb{Z}_p$ $\bar{h}_v = g^{a_1}$ $\bar{u} = g^{a_2}$ $\bar{v} = h^{a_2} g^{a_3}$ $u^* = g^{z_3}(u'/u)^{\omega_2}$ $v^* = h^{z_3}(v'/v)^{\omega_2}$	$\omega_1, z_1, z_2, z_3, a_4 \leftarrow \mathbb{Z}_p$ $\bar{h}_v = g^{z_1} {h_v}^{\omega_1}$ $\bar{u} = g^{z_2}(u')^{\omega_1}$ $\bar{v} = h^{z_2} g^{z3}(v')^{\omega_1}$ $u^* = g^{a_4}$ $v^* = h^{a_4}$
$\omega = H(x, \bar{h}_v, \bar{u}, \bar{v}, u^*, v^*)$	
$\omega_1 = \omega - \omega_2$ $z_1 = a_1 - \omega_1 \alpha$ $z_2 = a_2 - \omega_1 r$ $z_3 = a_3 - \omega_1 m$	$\omega_2 = \omega - \omega_1$ $z_4 = a_4 - \omega_2 r$
Prover sends $\pi = (\omega_1, \omega_2, z_1, z_2, z_3, z_4)$ to the verifier	

Figure 5: Prover steps for proving $\mathcal{R}_{\mathsf{Enc}}$ in DeVoS, excluding the $m \in C$ part, for $w = (\alpha, m, r)$ or $w = r$.

$\mathbb{G}^2$. To decrypt a ciphertext (u, v) w.r.t. secret key $\mathsf{sk} = (\mathbb{G}, p, g, \alpha)$, compute $m \leftarrow \log_g(v \cdot u^{-\alpha})$.

If the decisional Diffie-Hellman (DDH) problem is hard relative to $\mathbb{G}$, then (exponential) ElGamal is IND-CPA-secure. In order to re-randomize an ElGamal ciphertext (u, v), where $\mathsf{pk} = (\mathbb{G}, p, g, h)$ is the public key, choose $s \in \mathbb{Z}_p$ uniformly at random and return $(u', v') \leftarrow (u \cdot g^s, v \cdot h^s)$.

NIZKP of correct key generation. The NIZKP of correct key generation π_{KeyGen} is trivial, since for a given public key $(g, h) \in \mathcal{R}_{\mathsf{KeyGen}}$ we only need to check whether g and h are group members and g is a generator of $\mathbb{G}$.

One-way function. We choose modular exponentiation in $\mathbb{G}$ with respect to g, i.e., $f\colon \mathbb{Z}_p \to \mathbb{G}$ with $x \mapsto g^x$.

NIZKP of correct encryption. The specific relation to be proven by the voter V with key pair $(\mathsf{pk}_v, \mathsf{sk}_v) = (h_v = g^{\alpha_v}, \alpha_v)$ is:

$$\mathcal{R}_{\mathsf{Enc}} = \Big\{\Big(x = \big(\mathrm{e} = (u, v), \mathrm{e}' = (u', v'), g, h, h_v, C\big), w = (\alpha_v, m, r)\Big) : \\ (h_v = g^{\alpha_v}) \wedge (u' = g^r, v' = h^r g^m) \wedge (m \in C) \\ \vee (u' = u \cdot g^r, v' = v \cdot h^r)\Big\}.$$

Our proof system is based on Schnorr signature [49] and existing techniques for proving relations between group elements [4, 6] implemented in bilinear groups of size 256 bits and range proofs for enforcing $m \in C$ [6, 13, 17].

Considering the hardness of the DDH problem in group $\mathbb{G}$, the following protocol is a zero-knowledge proof of knowledge proof system for $\mathcal{R}_{\mathsf{Enc}}$, excluding the $m \in C$. We discuss the $m \in C$ part in App. F.2.

- The prover steps for $\mathcal{R}_{\mathsf{Enc}}$, excluding the $m \in C$ part, are shown in Fig. 5.
- The verifier outputs 1 on inputs (x, π) if and only if:

$$\omega_{\mathsf{V}} = H(x, g^{z_1} {h_v}^{\omega_1}, g^{z_2}(u')^{\omega_1}, g^{z_3}(u'/u)^{\omega_2}, h^{z_3}(v'/v)^{\omega_2}) \\ \omega_{\mathsf{V}} \stackrel{?}{=} \omega_1 + \omega_2$$

NIZKP of shuffle and of correct decryption. Since there are several ways to instantiate the NIZKP of shuffle π_{Shuffle} for ElGamal PKE in the literature (e.g., [30, 52]), we will not single out one of them. For the NIZKP of correct decryption, we can choose a standard Schnorr proof [49].

F INSTANTIATION OF DEVOS WITH PRACTICAL EVERLASTING PRIVACY

We employ the gluing technique from [26] for proving the public and private relations in zero-knowledge at the same time, which allows us to avoid the duplicate work when proving the relations separately. The idea is to use a nested hash function in the commit phase of the sigma protocol, where the terms related to the private relation are hashed first and their hash value is used as input to the outer hash function to hash the remaining terms w.r.t. the public relation. To fully hide information associated with the inner hash value, a perfectly hiding commitment to this value is used instead. We now describe NIZKPs π_{Enc} and π_{Com} for relations $\mathcal{R}_{\mathsf{Enc}}$ and $\mathcal{R}_{\mathsf{Com}}$ in DeVoS^{EL}, excluding the $m \in C$. We discuss the $m \in C$ part in App. F.2.

F.1 NIZKPs of correct encryption and correct commitment

For a proof of correct encryption π_{Enc}, we first describe $\mathcal{R}_{\mathsf{Enc}}$, which can only be verified by the parties who have access to SBB and PBB. The relation $\mathcal{R}_{\mathsf{Enc}}$ is as follows:

$$\mathcal{R}_{\mathsf{Enc}} = \Big\{(x, w) = \Big(x = \big(\mathrm{e} = (e_1, e_2, e_3, c_1, c_2), \\ \mathrm{e}' = (e_1', e_2', e_3', c_1', c_2'), g, h, h_1, g_1, g_2, h_v, C\big), \\ w = (r_1, r_2, r_3, \alpha_v)\Big) : \\ (h_v = g^{\alpha_v}, e_1' = g^{r_2}, e_2' = g^{r_3}, e_3' = g_1^{r_1} g_2^{r_3}, c_1' = h^{r_1} h_1^{r_2}, \\ c_2' = g_1^{r_2} g^m) \wedge (m \in C) \\ \vee (e_1' = e_1 g^{r_2}, e_2' = e_2 g^{r_3}, e_3' = e_3 g_1^{r_1} g_2^{r_3}, \\ c_1' = c_1 h^{r_1} h_1^{r_2}, c_2' = c_2 g_1^{r_2})\Big\}.$$

For public verifiability, we need a *NIZKP of correct commitment* π_{Com} with the underlying relation $\mathcal{R}_{\mathsf{Com}}$. We now describe the public relation $\mathcal{R}_{\mathsf{Com}}$.

$$\mathcal{R}_{\mathsf{Com}} = \Big\{(x, w) = \Big(x = \big(\mathrm{c} = (c_1, c_2), \mathrm{c}' = (c_1', c_2'), \\ g, h, h_1, g_1, g_2, h_v, C\big), w = (r_1, r_2, \alpha_v)\Big) : \\ (h_v = g^{\alpha_v}, c_1' = h^{r_1} h_1^{r_2}, c_2' = g_1^{r_2} g^m)) \wedge (m \in C) \\ \vee (c_1' = c_1 h^{r_1} h_1^{r_2}, c_2' = c_2 g_1^{r_2})\Big\}.$$

The proof system is based on the same ingredients as we mentioned previously for DeVoS (see App. E).

- Prover steps for $w = (r_1, r_2, r_3, \alpha)$ or $w = (r_1, r_2, r_3)$ in Fig. 6.
- The verifier of π_{Com} outputs 1 on inputs $(x_{\mathsf{PBB}}, \mathrm{c}, \mathrm{c}', \pi_{\mathsf{PBB}})$ if and only if: $\omega_{\mathsf{V}} = H_2(c_{\mathsf{SBB}}, x_{\mathsf{PBB}}, g^{z_1} h_v^{\omega_1}, h^{z_4} h_1^{z_2}(c_1')^{\omega_1}, h^{z_7} h_1^{z_5} \cdot (c_1'/c_1)^{\omega_2}, g_1^{z_5}(c_2'/c_2)^{\omega_2})$ and $\omega_{\mathsf{V}} \stackrel{?}{=} \omega_1 + \omega_2$.
- The verifier of π_{Enc} outputs 1 on inputs $(x_{\mathsf{PBB}}, x_{\mathsf{SBB}}, \mathrm{e}, \mathrm{e}', \pi_{\mathsf{PBB}}, \pi_{\mathsf{SBB}})$ if and only if it verifies on public inputs as described above and:

$w = (r_1, r_2, r_3, \alpha)$	$w = (r_1, r_2, r_3)$
$\omega_2, a_1, \ldots, a_4, z_5, z_6, z_7 \leftarrow \mathbb{Z}_p$	$\omega_1, z_1, z_2, z_3, z_4, a_5, a_6, a_7 \leftarrow \mathbb{Z}_p$
$\bar{h}_v = g^{a_1}$	$\bar{h}_v = g^{z_1} h_v^{\omega_1}$
$\bar{e}_1 = g^{a_2}$	$\bar{e}_1 = g^{z_2} (e'_1)^{\omega_1}$
$\bar{e}_2 = g^{a_3}$	$\bar{e}_2 = g^{z_3} (e'_2)^{\omega_1}$
$\bar{e}_3 = {g_1}^{a_4} {g_2}^{a_3}$	$\bar{e}_3 = g_1^{z_4} g_2^{z_3} (e'_3)^{\omega_1}$
$\bar{c}_1 = h^{a_4} {h_1}^{a_2}$	$\bar{c}_1 = h^{z_4} h_1^{z_2} (c'_1)^{\omega_1}$
$\bar{c}_2 = g_1^{a_2} g^{a_?}$	
$e_1^* = g^{z_5} (e'_1/e_1)^{\omega_2}$	$e_1^* = g^{a_5}$
$e_2^* = g^{z_6} (e'_2/e_2)^{\omega_2}$	$e_2^* = g^{a_6}$
$e_3^* = g_1^{z_7} g_2^{z_6} (e'_3/e_3)^{\omega_2}$	$e_3^* = g_1^{a_7} g_2^{a_6}$
$c_1^* = h^{z_7} h_1^{z_5} (c'_1/c_1)^{\omega_2}$	$c_1^* = h^{a_7} h_1^{a_5}$
$c_2^* = g_1^{z_5} (c'_2/c_2)^{\omega_2}$	$c_2^* = g_1^{a_5}$
$x_{\mathsf{SBB}} = (e_1, e_2, e_3, e'_1, e'_2, e'_3)$ $x_{\mathsf{PBB}} = (g, h, g_1, g_2, h_v, c_1, c_2, c'_1, c'_2)$ $\omega' = H_1(x_{\mathsf{SBB}}, \bar{e}_1, \bar{e}_2, \bar{e}_3, e_1^*, e_2^*, e_3^*)$ $(c_{\mathsf{SBB}}, a_{\mathsf{SBB}}) = commit(\omega')$ $\omega = H_2(c_{\mathsf{SBB}}, x_{\mathsf{PBB}}, \bar{h}_v, \bar{d}_1, d_1^*, d_2^*)$	
$\omega_1 = \omega - \omega_2$	$\omega_2 = \omega - \omega_1$
$z_1 = a_1 - \omega_1 \alpha$	$z_5 = a_5 - \omega_2 r_2$
$z_2 = a_2 - \omega_1 r_2$	$z_6 = a_6 - \omega_2 r_3$
$z_3 = a_3 - \omega_1 r_3$	$z_7 = a_7 - \omega_2 r_1$
$z_4 = a_4 - \omega_1 r_1$	
Prover sends $\pi_{\mathsf{PBB}} = (d_{\mathsf{SBB}}, \omega_1, \omega_2, z_1, z_2, z_4, z_5, z_7)$ to PBB and $\pi_{\mathsf{SBB}} = (a_{\mathsf{SBB}}, z_3, z_6)$ to SBB	

Figure 6: Prover steps for proving $\mathcal{R}_{\mathsf{Enc}}$ and $\mathcal{R}_{\mathsf{Com}}$ in DeVoS^{EL} excluding the $m \in C$ part, for $w = (r_1, r_2, r_3, \alpha)$ or $w = (r_1, r_2, r_3)$.

$$\omega'_{\mathsf{V}} = H_1(x_{\mathsf{SBB}}, g^{z_2}(e'_1)^{\omega_1}, g^{z_3}(e'_2)^{\omega_1}, g_1^{z_4} g_2^{z_3}(e'_3)^{\omega_1}, g^{z_5}(e'_1/e_1)^{\omega_2}, g^{z_6}(e'_2/e_2)^{\omega_2}, g_1^{z_7} g_2^{z_6}(e'_3/e_3)^{\omega_2}) \text{ and } open(\omega'_{\mathsf{V}}, a_{\mathsf{SBB}}, e_{\mathsf{SBB}}) \overset{?}{=} 1.$$

F.2 Range proofs

Above and in App. E, we presented NIZKPs for $\mathcal{R}_{\mathsf{Enc}}$ and $\mathcal{R}_{\mathsf{Com}}$, excluding the $m \in C$ part. To account for the $m \in C$ part, one can seemingly integrate sigma protocols for one of the existing techniques for range proofs, e.g., the classical one [6, 13], for which a number of optimizations is known (e.g., [17]), or the log based [17]. Regardless of the tallying method, ciphertexts are represented as sequences $\mathsf{e}_1, ..., \mathsf{e}_\tau$ that encrypt bits $b_1, ..., b_\tau$. If the votes are tallied homomorphically, e_τ is computed from $\mathsf{e}_1, ..., \mathsf{e}_{\tau-1}$, which poses a constraint $\sum_{i=1}^{\tau-1} b_i \leq 1$.

G DIFFERENTIAL PRIVACY

In general, a DP mechanism reveals generic information about a dataset (e.g., by publishing statistics of it) while concealing information about individuals in it [19]. The core idea originates from the cryptographic notion of indistinguishability: it ensures that an arbitrary change on a single record in the database has a negligible (bounded) effect on the result. The most common way to satisfy this is by injecting noise (e.g, Laplacian or Gaussian) into the underlying mechanism. Nowadays, DP is considered the flagship of data privacy definitions with several relaxations [16], some of which were adopted by organizations such as US Census Bureau [24], Microsoft [18], and Google [20]. DP is formalized in Def. G.1.

Definition G.1 (ε-differential privacy [19]). A privacy mechanism $\mathcal{M}$ is ε-differential private (or ε-DP) if

$$\mathbb{P}[\mathcal{M}(D_1) \in S] \leq e^{\varepsilon} \cdot \mathbb{P}[\mathcal{M}(D_2) \in S] \quad (1)$$

holds for all datasets D_1 and D_2 that differ only in one record and for all measurable set of possible results S.

In contrast, OSDP [37] changes this definition by ... *for all records D_1 and D_2, where D_2 is created by arbitrarily changing an existing record in D_1*".

G.1 OSDP Proof

Proof of Theorem 7.1. Assume there are n voters with votes $D = \{v_1, \ldots, v_n\}$ where v_i can be a real vote or $\perp$, i.e., a missing one. The output of $\mathcal{M}$ is $\{b_1, \ldots, b_n\}$ where b_i is either a legitimate or a fabricated ballot (they are indistinguishable) or $\perp$. Due to the IID nature of the added noise, the following holds where S and S_i is the set of possible ballots values for all and for a specific voter, respectively.

$$\Pr[\mathcal{M}(D) \in S] = \Pr[\mathcal{M}(v_1, \ldots, v_n) \in \{S_1, \ldots, S_n\}] \quad (2)$$
$$= \Pr[\mathcal{M}(v_1) \in S_1] \cdot \cdots \cdot \Pr[\mathcal{M}(v_n) \in S_n] \quad (3)$$

Due to this independence of the voters and their dummy ballots, we can simplify and focus on datasets with single voters and disregard the rest. The two possible sets of outputs that the adversary should not be able to differentiate is whether the voter casts a vote. Since the latter case is not sensitive, Eq. 4 does not need to be satisfied. On the other hand, if the voter casts a vote, then Eq. 5 and 6 must be satisfied where $\hat{S}$ = {ballot recorded for voter} and $\tilde{S}$ = {ballot not recorded for voter}. Note that the definition of OSDP does not specify the sensitivity of the substituting record, i.e., Eq. 5 and 6 should hold with "voter voted *for another candidate*" besides "voter not voted" as well, which is indeed the case due to the properties of DeVoS.

$$\mathbb{P}[\mathcal{M}(\text{voter not voted}) \in S] \leq e^{\varepsilon} \cdot \mathbb{P}[\mathcal{M}(\text{voter voted}) \in S] \quad (4)$$
$$\mathbb{P}\left[\mathcal{M}(\text{voter voted}) \in \hat{S}\right] \leq e^{\varepsilon} \cdot \mathbb{P}\left[\mathcal{M}(\text{voter not voted}) \in \hat{S}\right] \quad (5)$$
$$\mathbb{P}\left[\mathcal{M}(\text{voter voted}) \in \tilde{S}\right] \leq e^{\varepsilon} \cdot \mathbb{P}\left[\mathcal{M}(\text{voter not voted}) \in \tilde{S}\right] \quad (6)$$

When we reformulate these equations with the corresponding probabilities, we get $1 \leq e^{\varepsilon} \cdot p$ and $0 \leq e^{\varepsilon} \cdot (1 - p)$, respectively. While the second holds trivially, the first corresponds to the formula in Theorem 7.1. □

G.2 Approximated DP

In the following, we show why the vanilla ε-DP is not satisfied by using strategy $\boxed{3}$ if $p < 1$. What is more, we show that even (ε, δ)-DP — a more general DP relaxation which includes a small additive term $\delta > 0$ at the end of Eq. 1 — could only be satisfied with extremely weak privacy parameters due to the asymmetry.[13]

Theorem G.2. *Suppose* PT *generates each dummy ballot for each voter following IID* Bernoulli(p) *distribution where $p < 1$ (i.e., via*

[13] A similar result can be produced for strategy $\boxed{4}$ as well.

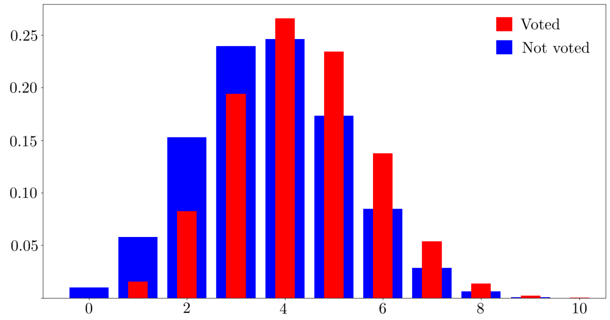

Figure 7: Two output distribution of the number of ballots when $p = 0.37$ after ten submission phase. Blue and red corresponds to one vote and no vote 'after coercion' respectively.

strategy $\boxed{3}$), then DeVoS does not satisfy ε-DP for any $\varepsilon \in \mathbb{R}$. Furthermore, (ε, δ)-DP could only be satisfied with meaningless privacy parameters when $1 - p \le \delta$.

Proof of Theorem G.2. Following the proof of Theorem 7.1, because DP is symmetric (in contrast to OSDP), it is sufficient to show that Eq. 1 is violated for a dataset with a single voter. Similarly to Eq. 4, if $D_1 = \{\perp\}$, $D_2 = \{v\}$ and $S = \{\text{no ballot was recorded}\}$, then on the left side of Eq. 7 the probability is $(1 - p)$ while on the right it is zero which is a contradiction if $p \neq 1$. Thus, DeVoS only satisfies DP when there is a ballot (fabricated dummy or legit) for all voters.

$$1 - p = \mathbb{P}\left[\mathcal{M}(D_1) \in S\right] \le e^{\varepsilon} \cdot \mathbb{P}\left[\mathcal{M}(D_2) \in S\right] = 0 \tag{7}$$

Considering the relaxed (ε, δ) -DP, the additive term δ could cover the above case when $1 - p \le \delta$. However, in practice, δ should be extremely small as it should only capture the sporadic cases. Indeed, the best practice is to set δ to the inverse of the size of the dataset. This could only be achieved by setting p very close to 1, which does not offer any meaningful improvement over the original DeVoS protocol. □

G.3 Multi-submission phases

Proof of Theorem 7.2. The guarantee OSDP provides is Eq. 1, i.e., the probability distribution (when no vote is cast) on the right side must be scaled up sufficiently (with e^{ϵ}) to cover the probability distribution (when c vote is cast) on the left side. The probabilities of the number of recorded ballots are illustrated in Fig. 7 after ten submissions when $c = 1$ and $p \approx e^{-1}$ where blue and red distributions are the left and right, respectively.

More generally, when c votes are cast during m submission phase, the probabilities for each output are shown in Table 4. The first column is the number of recorded ballots, and the second and third are the events' probabilities when 0 and c votes were cast, respectively.

Focusing on the probability parts in one row, we can see that to scale the second column (i.e., blue in Fig. 7) above the third one (i.e., red in Fig. 7), one must multiply the former with $p^{-c} (= e^{\epsilon})$ element-wise.

Focusing on the binomial parts in one row, we can see in Eq. 8, that the expression in the second column is greater (or equal) than the one in the third when $i = \{0, 1, \ldots, m - c\}$. Moreover, the difference is the largest ($\binom{m}{c}$ times) when $i = 0$, and the smallest (equal size) when $i = m - c$.

$$\binom{m}{c+i} = \binom{m-c}{i} \cdot \frac{m \cdots (m-c+1)}{(c+i) \cdots (i+1)} \tag{8}$$

Thus, due to the last row when the binomial parts are equal, one must set $\epsilon = \log\left(\frac{1}{p^c}\right) = c \cdot \log\left(\frac{1}{p}\right)$ to satisfy Eq. 1. □

G.4 Strategy $\boxed{4}$

Theorem G.3. *If we assume q fraction of voters vote in a submission phase and if* PT *generates dummy ballots for a uniformly randomly selected $p' - q$ voters who do not vote in the current submission phase, then strategy $\boxed{4}$ satisfies $\log\left(\frac{N\cdot(1-\hat{q})+1}{N\cdot(p'-\hat{q})+1}\right)$-OSDP within a submission phase, where the presence of a ballots is protected and $\hat{q}$ is the potential maximum fraction of active voters in a submission phase.*

Proof. The proof is analogous with the proof presented in for strategy $\boxed{3}$. Due to the independence of votes, it is sufficient to focus on a single voter with vote v and ballot b. However, contrary to strategy $\boxed{3}$, the probability of generating a dummy ballot for this particular voter is not independent of the other voters: when q voters voted, $\Pr\left[b \neq \perp | v = \perp\right] = \frac{N\cdot p' - N\cdot q}{N - N\cdot q}$. Substituting this formula into Eq. 5 we get the left side of Eq. 9. As this should hold even for the worst case (i.e., when maximum votes are caster including v), we can change q with $\frac{N\cdot\hat{q}-1}{N}$, so the lower bound is the highest. After some arithmetic, we can see that the right side of Eq. 9 is the same as the desired formula in the theorem.

$$\log\left(\frac{1-q}{p'-q}\right) \le \varepsilon \Rightarrow \log\left(\frac{1-\frac{N\cdot\hat{q}-1}{N}}{p'-\frac{N\cdot\hat{q}-1}{N}}\right) \le \varepsilon \tag{9}$$

□

We leave to extend this result to multi-submission phases as future work.

Table 4: Two output distribution of the number of ballots: the first column is the number of recorded ballots while the second and the third column are the probabilities of that event when 0 and c votes were casted respectively.

Num. of ballots	Pr. with 0 votes	Pr. with c votes
0	$\binom{m}{0} \cdot (1-p)^{m}$	0
$\vdots$	$\vdots$	$\vdots$
$c-1$	$\binom{m}{c-1} \cdot p^{c-1} \cdot (1-p)^{m-c+1}$	0
c	$\binom{m}{c} \cdot p^{c} \cdot (1-p)^{m-c}$	$\binom{m-c}{0} \cdot (1-p)^{m-c}$
$c+1$	$\binom{m}{c+1} \cdot p^{c+1} \cdot (1-p)^{m-c-1}$	$\binom{m-c}{1} \cdot p \cdot (1-p)^{m-c-1}$
$\vdots$	$\vdots$	$\vdots$
m	$\binom{m}{m} \cdot p^{m}$	$\binom{m-c}{m-c} \cdot p^{m-c}$