SoK: Secure e-voting with everlasting privacy

Thomas Haines
Australian National University
Canberra, Australia
thomas.haines@anu.edu.au

Rafieh Mosaheb
University of Luxembourg
Esch-sur-Alzette, Luxembourg
rafieh.mosaheb@uni.lu

Johannes Müller
University of Luxembourg
Esch-sur-Alzette, Luxembourg
johannes.mueller@uni.lu

Ivan Pryvalov
University of Luxembourg
Esch-sur-Alzette, Luxembourg
ivan.pryvalov@uni.lu

ABSTRACT

Vote privacy is a fundamental right, which needs to be protected not only during an election, or for a limited time afterwards, but for the foreseeable future. Numerous electronic voting (e-voting) protocols have been proposed to address this challenge, striving for *everlasting privacy*. This property guarantees that even computationally unbounded adversaries cannot break privacy of past elections.

The broad interest in secure e-voting with everlasting privacy has spawned a large variety of protocols over the last three decades. These protocols differ in many aspects, in particular the precise security properties they aim for, the threat scenarios they consider, and the privacy-preserving techniques they employ. Unfortunately, these differences are often opaque, making analysis and comparison cumbersome.

In order to overcome this non-transparent state of affairs, we systematically analyze all e-voting protocols designed to provide everlasting privacy. First, we illustrate the relations and dependencies between all these different protocols. Next, we analyze in depth which protocols do provide secure and efficient approaches to e-voting with everlasting privacy under realistic assumptions, and which ones do not. Eventually, based on our extensive and detailed treatment, we identify which research problems in this field have already been solved, and which ones are still open. Altogether, our work offers a well-founded reference point for conducting research on secure e-voting with everlasting privacy as well as for future-proofing privacy in real-world electronic elections.

KEYWORDS

electronic voting, everlasting privacy, protocol analysis

1 INTRODUCTION

Electronic voting (e-voting) systems are regularly used in real-world elections. Many countries, including Australia, Estonia, India, Switzerland, and the US, employ e-voting for political elections. Moreover, numerous institutions, for example IACR, ACM, or SIAM, make use of e-voting to enable all members to participate, regardless of their physical locations.

In all elections, it is crucial to ensure that the final election result correctly reflects the votes chosen by the voters. Moreover, voters' individual votes must remain secret so that the final result is not biased by those who are afraid to express their own will freely. In order to guarantee these two fundamental properties, modern *secure* e-voting protocols strive for (end-to-end) verifiability and (vote) privacy. *Verifiability* [19] enables external and internal observers to verify whether the final election result corresponds to the voters' choices, even when some participants (e.g., talliers) are malicious and try to undetectably manipulate the final outcome. *Privacy* [5] ensures that all data published during an election (including data for proving the correctness of the final result) does not leak more information on the single voters' choices than what can be derived from the public election result.

For many elections, it is important to protect voters' privacy not only during the election, or for a limited time afterwards, but also for the foreseeable future. If voters need to worry about facing negative consequences in case their individual votes are leaked, say, 10 or 20 years after the election, then this fear can undermine the integrity of the final result. This threat is real, both for political and for non-political elections. For instance, democratic systems, in which representatives are elected by the public, can be overthrown and be replaced by oppressive regimes which discriminate or even prosecute people who supported its opponents in the past. But also in non-political elections, the choices that voters make now can still be sensitive several years later, for example in an election of a university's president.

In classical paper-based elections, long-term privacy is commonly protected by the same mechanisms that also ensure privacy during the election. But for electronic voting, the situation is different. In order to guarantee verifiability (see above), some information about the voters' individual choices needs to be public. Since, at the same time, vote privacy must not be jeopardized, essentially all verifiable e-voting systems used in practice today (e.g., Helios [1] or Belenios [20]) employ the following approach: voters encrypt their votes under the talliers' public key, publish the resulting ciphertexts, and the talliers use their secret key to process these ciphertexts to obtain the final result. Now, the problem is that secrecy of all public-key encryption schemes deployed in these systems (e.g., ElGamal) is based on certain computational hardness assumptions (e.g., decisional Diffie-Hellman) that ensure vote privacy at the time of the election, but not necessarily in the long run. A future adversary, who learns from public data of past elections which ciphertext belongs to which voter, may therefore exploit novel (previously

Proceedings on Privacy Enhancing Technologies 2023(1), 279–293

https://doi.org/10.56553/popets-2023-0017

unknown) algorithms or more powerful machines (e.g., quantum computers) to efficiently solve the underlying hardness assumptions and thus break privacy of voters retrospectively. As explained above, such a risk is unacceptable for many real-world elections. However, many secure e-voting protocols, such as Helios [1], Civitas [18], Selene [64], sElect [47], Demos [44], or Ordinos [46], were not designed to protect against this threat.

Fortunately, in order to ensure that vote privacy remains preserved in the future, numerous e-voting protocols have been proposed in the academic literature (e.g., [11, 21, 22, 51, 56]). These protocols strive for what is called *everlasting privacy*. This property ensures that privacy is protected *unconditionally* so that even a computationally unbounded adversary is not able to learn how individual voters voted. Most of the e-voting protocols mentioned above actually aim for a weaker notion of everlasting privacy. In fact, these protocols are designed to guarantee unconditional privacy towards any external adversary who can access all public election data but who is not able to monitor the whole communication network. This relaxed notion of everlasting privacy is called *practical everlasting privacy* [2]. It accurately models the overall threat scenario of a future adversary who knows all public material required to verify an election and who is able to break any computational hardness assumption.

The diversity of e-voting protocols aiming for (practical) everlasting privacy offers great potential but it also poses a major challenge at the same time. The reason is that existing protocols differ in many aspects:

- *Security properties*: While some protocols were designed to guarantee "only" public verifiability and (everlasting) privacy (e.g., [21, 51]), other protocols aim to provide additional security properties, such as *receipt-freeness* (e.g., [56]), or *accountability* (e.g., [22]).
- *Threat scenarios*: Existing protocols often differ in the assumptions that they (sometimes implicitly) make to provide specific security properties. For example, some protocols aim to not only provide everlasting privacy towards external but also against internal adversaries (e.g., [11]).
- *Privacy-preserving techniques*: There exists a multitude of techniques that the protocols proposed employ to protect vote privacy or additional privacy-related features like *coercion-resistance*. While some protocols employ only one technique (e.g., [22]), others use two or more of them (e.g., [56]). In many cases, it is not explained which technique is supposed to provide which privacy-related property precisely.

These differences are often opaque, making analysis and comparison cumbersome. This results in a confusing situation that raises several fundamental questions:

(1) How do all the different e-voting protocols aiming for (practical) everlasting privacy relate? How do they depend on each other?
(2) Which of these protocols do provide secure and efficient solutions to e-voting with (practical) everlasting privacy under realistic assumptions? Which ones do not?
(3) Which research problems in this field have already been solved? Which ones are still open?
(4) Do there exist secure solutions that can be deployed to guarantee (practical) everlasting privacy in real-world electronic voting? If not, which gaps need to be closed?

Addressing these questions is crucial because the need for future-proofing privacy in electronic voting is pressing.

1.1 Our contributions

In order to overcome this non-transparent state of affairs, we answer all of the fundamental questions raised above. We do this in a systematic, critical, and detailed manner. As a result, our work offers a well-founded reference point for conducting research on secure e-voting with everlasting privacy as well as for future-proofing privacy in real-world electronic elections.

In what follows, we explain our approach and then describe our key findings. Before that, we clarify the scope of our work.

1.1.1 Scope. In this work, we strictly focus on systematizing *secure electronic* voting protocols with *everlasting* privacy that were published in the literature. In particular, we do not study purely paper-based voting protocols (e.g., [67]), e-voting protocols with post-quantum yet conditional privacy (e.g., [9, 10]), non-verifiable e-voting protocols (e.g., [72]), or generic information-theoretically secure MPC (e.g., [23]).

Moreover, we restrict our attention to *concrete constructions* of secure e-voting protocols with everlasting privacy, leaving out studies on the theoretical limits of secure e-voting (or more generally: secure MPC) with everlasting privacy (e.g., [17, 38, 59]).

1.1.2 Approach. We use the following approach to systematically analyze the state-of-the-art in secure e-voting with everlasting privacy:

(1) We study the academic literature to find all relevant existing protocols in this field.
(2) We classify existing protocols according to how they (intend to) provide everlasting privacy technically. Moreover, we illuminate how different protocols depend on each other.
(3) We analyze which existing protocols are practically efficient and guarantee public verifiability as well as (practical) everlasting privacy under realistic assumptions. To this end, we investigate which protocols actually achieve the properties they were designed for originally, and we critically reflect on the assumptions that existing protocols make.
(4) Based on our analysis in the previous steps, we identify which research problems have already been solved and which ones are still open.

1.1.3 Key insights. We state the main insights of our endeavor next. We start with the list of relevant protocols that we collected and then summarize our classification of these protocols. Afterwards, we explain which challenges have been solved and which ones are still open.

Existing protocols. We collected 25 existing e-voting protocols designed for secure e-voting with everlasting privacy [2, 11, 20–22, 25–27, 29, 30, 33–35, 40, 42, 45, 51–53, 55, 56, 61, 62, 69, 70].[1]

[1]We thank the anonymous reviewer who pointed out that we had omitted Helios [1] without identities, as described in [2], and Belenios [20].

Classification. We identify two different classes of existing protocols, B-ANON and B-ID. In B-ANON, everlasting privacy reduces to publishing ballots anonymously. On the contrary, in B-ID, where public ballots are identifiable, everlasting privacy is based on the privacy-preserving technique to tally ballots. We will argue that the general approach taken in B-ID is superior to the one in B-ANON; in short: B-ID > B-ANON (see Sec. 3.3).

Solved problems. We discover that in both classes, B-ID and B-ANON, there exist reasonable protocols for secure e-voting with everlasting privacy under the respective assumptions made in these classes. For everlasting privacy, all of these protocols consider future adversaries that are not active during an election. We distinguish between those protocols that can handle simple ballot types (e.g., where voters can choose one candidate) and those which can handle arbitrary ballot types (e.g., where voters can rank candidates).

Observation 1 (Simple ballot types). *In B-ID, there exist two secure approaches that can handle simple ballot types: the one based on [21] (see Sec. 6) and the one based on the homomorphic version of [22] (see Sec. 7). While [22] offers everlasting privacy towards the public (i.e, practical everlasting privacy), [21] additionally offers everlasting privacy towards a threshold of talliers.*

Observation 2 (Arbitrary ballot types). *In B-ID, there exists one secure approach that can handle arbitrary ballot types, the one based on the mix net version of [22] (see Sec. 7). In B-ANON, there exist two reasonably secure approaches that can handle arbitrary ballot types, in fact [20] (see Sec. 4) and [51] (see Sec. 5). These protocols offer practical everlasting privacy.*

All of the approaches mentioned before are sufficiently efficient for large-scale elections. In particular, Belenios [20] has already been deployed in many real-world elections.

Open problems. Our first two observations demonstrate that almost all of the main challenges have been solved, but there still exist some open problems.

Observation 3 (Open problems). *The most important open problems are:*

(1) Formal protocol analysis: *While the cryptographic components of the promising approaches [21, 22, 51] have been analyzed in-depth, it is an open problem to formally analyze these proposals on the* protocol *level. It is also an open problem to formally analyze everlasting privacy of Belenios [51].*

(2) Deployable e-voting system: *While Belenios [20], which is in B-ANON, can be deployed for real-world elections, it is an open problem to develop a full-fledged deployable e-voting system that realizes one of the promising approaches [21, 22] in the superior class B-ID.*

(3) Weaker trust for arbitrary ballot types: *All promising approaches that can handle arbitrary ballot types [20, 22, 51] require that all election authorities or all talliers are trusted for everlasting privacy. It is an open problem to mitigate trust on the authorities in terms of everlasting privacy for arbitrary ballot types.*

(4) Receipt-freeness: *In all of the promising approaches [20–22, 51], some evidence is created on the voters' devices that can serve as a proof for how the voter voted. It is an open problem to securely and efficiently improve [20–22, 51] so that they are free of such receipts.*

Table 1: Notation for e-voting protocols.

Variable	Meaning
EA	election authority
$\mathsf{V}_1, \ldots, \mathsf{V}_n$	voters
v_i	V_i's vote
ρ	result function
$\mathsf{T}_1, \ldots, \mathsf{T}_m$	trustees
$\mathsf{M}_1, \ldots, \mathsf{M}_l$	mix servers
PBB	public bulletin board
SBB	secret bulletin board

From our point of view, the first two open problems (formal protocol analysis and development of a deployable system in B-ID) are the most pressing ones. We note that for automated verification, there exist appropriate symbolic definitions to address the first open problem, for example [2] for everlasting privacy and [57] for verifiability/accountability; recent advances [16] facilitate applying these definitions in a joint verification platform.

1.2 Overview of paper

We structure our paper as follows. In Sec. 2, we recall the main principles of secure e-voting protocols; moreover, we introduce our notation. In App. A, we describe those cryptographic primitives which are commonly used to design secure e-voting protocols. In Sec. 3, we propose our classification and discuss which of the two main classes, B-ANON and B-ID, offers a more reasonable approach. We will further demonstrate in Sec. 3 that both main classes have two sub-classes each. The subsequent sections are dedicated to the protocols in these sub-classes, in fact Sec. 4 to B-ANON-A, Sec. 5 to B-ANON-V, Sec. 6 to B-ID-HOM, and Sec. 7 to B-ID-MIX. In these sections, we illustrate how existing protocols in the respective sub-classes relate, and we analyze which of them are actually reasonable approaches for secure e-voting with everlasting privacy. We summarized our insights in Table 3. We conclude in Sec. 8.

2 SECURE E-VOTING IN A NUTSHELL

The following section serves two purposes. First, we briefly introduce those readers to secure e-voting who are not (yet) familiar with the subject. Second, we determine a clear and unified notation for expressions that we will recurrently use in the main part of our paper.

We start off with explaining the basic model of e-voting. After that, we recall the most relevant properties of secure e-voting.

We refer to Table 1 for our notation related to e-voting and to Table 2 for our notation of cryptographic primitives.

2.1 Electronic voting

A voting protocol is run between an *election authority* EA, a set of *voters* $\mathsf{V}_1, \ldots, \mathsf{V}_n$, and a *trustee* T (sometimes called *tallier*). The election authority EA is responsible for setting up the election (date, set of candidates, voting method, etc.) and for registering voters.

During the submission phase, the voters $\mathsf{V}_1, \ldots, \mathsf{V}_n$ cast their individual votes $v_1, \ldots, v_n$. In the tallying phase, the trustee T then takes these votes as input, applies the specified voting method ρ to these votes (e.g., counts the number of votes per candidate), and eventually outputs the election result $\rho(v_1, \ldots, v_n)$. In *electronic* voting, voters' choices are encoded digitally and processed electronically.

It is obvious that, without any further measures, the trustee T needs to be trusted in all important aspects: first, T learns how all voters voted, and second, if T is dishonest, it can manipulate the election outcome undetectably. In order to avoid reliance on a single completely trusted authority, *secure* e-voting protocols offer and combine certain desirable properties, as we describe next.

2.2 Security properties

We recall the most important properties of secure e-voting. We start with the two fundamental ones (privacy and verifiability) and then explain the common high-level approach to combine them. Afterwards, we elaborate on three properties that several protocols strive for in addition, namely accountability, receipt-freeness, and coercion-resistance. We explain all security properties on an intuitive level, which is sufficient to follow our exposition, and we provide references to established formal definitions.

Basic security: privacy and verifiability. The two main requirements for secure e-voting, as mentioned in Sec. 1, are (vote) privacy [5] and public verifiability [19].

Privacy guarantees that the links between individual voters and their votes in the (public) final result remain secret. In order to mitigate trust on the trustee T for privacy, its role is often distributed among several entities $\mathsf{T}_1, \ldots, \mathsf{T}_m$ so that only a threshold of them need to be trusted for privacy. Moreover, depending on how trust is distributed, the protocol becomes more robust in case a trustee is not able (or willing) to participate in tallying. Furthermore, some protocols divide or distribute the role of EA among several parties; for example, Belenios distinguishes between the EA proper and the registrar who generates the voters' signing keys.

Public verifiability ensures that everyone is able to verify that the final election result is correct, even if the trustees or other participants are corrupted. Since the main purpose of verifiability is to protect against possibly corrupted parties, verifiability should (ideally) not be based on any trust assumptions. This is in contrast to privacy for which trust in some of the tallying authorities appears unavoidable (hence, the term "trustee(s)").

Basic security: common approach. Secure e-voting protocols commonly employ an additional party, called the *public bulletin board* PBB. The role of PBB is to broadcast all data that is necessary to verify the correctness of the final result.[2] During the submission phase, the voters $\mathsf{V}_1, \ldots, \mathsf{V}_n$ publish some information about their votes on PBB; commonly, in order to protect their own privacy, voters seal their votes v_i in secret ballots. In the tallying phase, the trustees $\mathsf{T}_1, \ldots, \mathsf{T}_m$ use some secret knowledge (e.g., private keys) to process the secret ballots on PBB and obtain the final result $res = \rho(v_1, \ldots, v_n)$.[3] Eventually, the trustees publish on PBB the result as well as some evidence that convinces everyone that res was computed honestly, but without revealing the trustees' secret knowledge. As we shall see in the remainder of our paper, there exist several techniques to realize this high-level approach, each one with its own advantages and disadvantages.

Accountability. Accountability is a stronger notion of verifiability [49]. While public verifiability enables everyone to verify whether the final election result correctly reflects the votes chosen by the voters, verifiability alone is not sufficient to detect which parties manipulated the final outcome. Accountability solves this problem as it enables one to identify misbehaving parties individually and hold these parties accountable. This property is particularly useful in practice because it serves as a deterrent.

Receipt-freeness. Receipt-freeness ensures that the casting process does not create any (cryptographic) evidence that proves how the voter voted [13]. While vote privacy guarantees that the overall election process does not reveal how individual voters voted, the final outcome can still be biased due to vote buying. In fact, the process executed by the voter to create and submit her ballot may leave some local data on the voter's device. A voter can sell this data as a receipt that proves how she voted. In receipt-free e-voting protocols, no convincing evidence is created on the voters' devices.

Coercion-resistance. Coercion-resistance protects against adversaries who coerce voters to follow certain instructions so that the adversary achieves his own goals (e.g., the voter votes for the coercer's favorite candidate) [48]. More precisely, a voting protocol is coercion-resistant if any coerced voter, instead of obeying the coercer, can run some counter-strategy such that (1) by running the counter-strategy, the coerced voter achieves her goal (e.g., successfully votes for her favorite candidate), and (2) the coercer is not able to distinguish whether the coerced voter followed his instructions or tried to achieve her own goal. Unlike receipt-freeness, coercion-resistance also considers voters who *actively* deviate from their prescribed roles.

3 OUR CLASSIFICATION

We propose a classification that captures all existing e-voting protocols aiming for everlasting privacy. Our classification is particularly useful to answer our initial research questions (Sec. 1). As we shall see, there exist two different main classes, each of which has two sub-classes. In Sec. 4–7, we elaborate on existing protocols in these sub-classes, where we focus on one sub-class in each section.

In this section, we first describe our classification and then explain how different classes relate from a high-level perspective. Eventually, we give some fundamental insights to the question which of the two main classes is more reasonable under realistic assumptions.

3.1 Classes

Based on our extensive literature research, we identify the following two main classes of e-voting protocols with everlasting privacy.

[2] Some e-voting protocols with everlasting privacy also employ a secret bulletin board SBB whose role is to share certain information only among the trustees.

[3] Some e-voting protocols additionally employ a set of mix servers $\mathsf{M}_1, \ldots, \mathsf{M}_l$ in the tallying phase.

Table 2: Notation for common cryptographic primitives (see App. A for background information).

Variable	Meaning
$\mathcal{S} = (\mathsf{KG}_s, \mathsf{S}, \mathsf{V}_s)$	digital signature scheme
(vk, ssk)	verification/secret signing key pair
$s \leftarrow \mathsf{S}(ssk, m)$	signature s on message m
$b \leftarrow \mathsf{V}_s(vk, m, s)$	verification of signature s on msg. m
$\mathcal{E} = (\mathsf{KG}_e, \mathsf{E}, \mathsf{D})$	public-key encryption scheme
(pk, sk)	public/private encryption key pair
$e \leftarrow \mathsf{E}(pk, m)$	encryption e of message m
$m' \leftarrow \mathsf{D}(sk, e)$	decryption m' of ciphertext e
$\mathcal{C} = (\mathsf{KG}_c, \mathsf{C}, \mathsf{O})$	commitment scheme
prm	commitment parameters
$c \leftarrow \mathsf{C}(prm, m, r)$	commitment c to message m
$b \leftarrow \mathsf{O}(prm, c, m, r)$	verif. of opening (m, r) for comm. c
$\Pi = (\mathsf{P}, \mathsf{V})$	non-interactive zero-knowledge proof
$\pi \leftarrow \mathsf{P}(s, w)$	proof for statement s using witness w
$b \leftarrow \mathsf{V}(s, \pi)$	verif. of proof π for statement s

Anonymous ballots (B-ANON). Here, the ballots published on the bulletin board do not contain any information that could be used to trace single ballots back to individual voters. Everlasting privacy is based on the assumption that the ballot submission channels are unconditionally anonymous towards the adversary. Existing protocols in B-ANON are [2, 20, 27, 33, 35, 40, 42, 45, 51–53, 62, 70].

We further subdivide B-ANON according to the party who generates the voters' credentials that the voters use to prove eligibility of their ballots anonymously:

- B-ANON-A: The election authority EA creates all voters' credentials. Existing protocols in B-ANON-A are [2, 20, 27, 33, 40, 42, 62, 70].
- B-ANON-V: Each voter creates her credentials by herself. Existing protocols in B-ANON-V are [35, 45, 51–53].

We study sub-class B-ANON-A in Sec. 4, where we demonstrate that all existing protocols in B-ANON-A put much trust in the election authority EA. We study the sub-class B-ANON-V in Sec. 5; we will see that the only secure and practically efficient protocol in this sub-class is [51], but under the arguably strong assumption that all ballot submission channels are unconditionally anonymous (see Sec. 3.3).

Identifiable ballots (B-ID). Here, the ballots published on the bulletin board contain some information (e.g., voters' IDs) which enables everyone to identify which ballot belongs to which voter. Everlasting privacy is based on the unconditionally privacy-preserving technique used to tally ballots. Existing protocols in B-ID are [11, 21, 22, 25, 26, 29, 30, 34, 55, 56, 61, 69].

We further subdivide B-ID according to the privacy-preserving technique that is used in the tallying phase:

- *Homomorphic tallying* (B-ID-HOM): The trustees first homomorphically aggregate the secret ballots published by the voters and then open the aggregated result. The outcome of this procedure contains the total number of votes for each candidate/choice. Existing protocols in B-ID-HOM are [21, 22, 26, 29, 69].
- *Mix-net tallying* (B-ID-MIX): The trustees shuffle all secret ballots published by the voters and then open the shuffled result. The outcome of this procedure contains all voters' choices in a permuted order. Existing protocols in B-ID-MIX are [11, 22, 25, 26, 30, 34, 55, 56, 61].

We note that the protocols in B-ID-MIX can handle more complex ballots than existing ones in B-ID-HOM.[4]

We study sub-class B-ID-HOM in Sec. 6, where we demonstrate that the best protocols in this sub-class are [21] and its extension [29], as well as the homomorphic version of [22]. We study sub-class B-ID-MIX in Sec. 7, where we show that the mixing version of [22] offers the most reasonable approach, including the works built upon it [30, 34, 61].

3.2 Relations

We observe that the two main classes B-ANON and B-ID essentially differ in two aspects: (1) the method used to ensure everlasting privacy as well as the phases when the respective method is applied, and (2) the technique employed to guarantee public verifiability. More precisely:

Privacy. In B-ANON, everlasting privacy follows from the property that the method used to publicly prove eligibility of ballots does not leak any information about the respective voters' identities, and from the assumption that the voters' submission channels are unconditionally anonymous towards the adversary. In contrast, in B-ID, everlasting privacy is ensured by employing an unconditionally privacy-preserving technique to process ballots, either homomorphic aggregation (B-ID-HOM) or mixing (B-ID-MIX) of ballots, and under the assumption that the voters' submission channels are unconditionally private towards the adversary.

In particular, in B-ANON, individual links are broken *before* ballots are published (submission phase), whereas in B-ID, these links are broken *afterwards* (tallying phase).

Verifiability. In B-ANON, verifiability essentially reduces to the technique that voters use to prove that their anonymous ballots were submitted by eligible voters. In contrast, in B-ID, verifiability mainly follows from the techniques used to prove that ballots were processed correctly in the tallying phase.

[4] The reason is that for homomorphic tallying, voters need to prove that their secret votes are in fact valid; otherwise, a dishonest voter could exploit the homomorphic property to secretly add or remove arbitrary numbers of votes. To this end, protocols in B-ID-HOM employ NIZKPs to prove and verify that each voter submitted a valid vote. However, the NIZKPs used in existing protocols in B-ID-HOM become practically inefficient for larger number of choices. (One option to mitigate this limitation is to realize these NIZKPs with Succinct Non-Interactive Arguments of Knowledge (SNARKs), as recently proposed in [39].) This is, for instance, the case when voters can choose between many different candidates or when voters can rank candidates (e.g., in instant-runoff voting). On the contrary, in B-ID-MIX, voters only need to prove knowledge of their secret votes in order to ensure that they created their ballots independently; this measure protects, in particular, against replay attacks that violate vote privacy [54]. The NIZKPs employed for this purpose are typically independent of the number of choices. Therefore, protocols in B-ID-MIX can handle arbitrarily complex ballots.

3.3 Basic evaluation

We found that the general approach taken in B-ID, where ballots are identifiable, is superior to the one in B-ANON, where ballots are necessarily anonymous. In short, loosely speaking, we claim:

$$\text{B-ID} > \text{B-ANON}$$

In what follows, we substantiate this statement. Since our argumentation is heuristic, for the sake of fairness, we will evaluate existing protocols in B-ANON (see Sec. 4-5) independently of our general criticism.

Recall that all protocols in B-ANON require an unconditionally *anonymous* submission channel to achieve everlasting privacy, whereas protocols in B-ID require an unconditionally *private* but not necessarily anonymous submission channel. As we will argue next, the gap between these two requirements is significant in practice.

Basic observation. We start with a simple but important fact: perfect secrecy is impossible to achieve in a public-key setting. Hence, unconditionally private or unconditionally anonymous channels can only be realized in a symmetric setting. Such a setting is, however, unrealistic in real-world Internet elections, which is the most dominant form of electronic voting nowadays. This means that, in practice, both unconditional privacy and unconditional anonymity of the submission channels cannot be based on actual cryptographic constructions; instead, these properties follow from the *assumption* that all adversaries considered are not able to break privacy or anonymity, respectively, of the submission channels.

In what follows, we argue that in B-ID the respective assumption can both be justified for a larger class of elections and be mitigated, but that neither holds in B-ANON.

Adversarial power. It is well-known that multiple agencies across the globe monitor the Internet and permanently store some of its traffic. Since storing all communication data is practically infeasible, these agencies are primarily interested in *metadata*. If we assume that a future adversary has/gains access to metadata collected during a past election, then anonymity of the voters' submission channels is violated. At the same time, privacy of these channels is still guaranteed under the assumption that only metadata but no further data was stored. We can therefore conclude that the class of potential future adversaries against which protocols in B-ID can protect is larger than the one that protocols in B-ANON can handle.

Mitigation techniques. Even though, as explained above, we cannot establish unconditionally private submission channels in real-world elections, there exist several techniques to realize channels with long-term secrecy, in particular post-quantum TLS (e.g., [66]) which can easily be deployed in real-world Internet elections, without any effort on the voters. For anonymous channels, the situation is different: anonymous communication protocols are currently used by only a tiny fraction of Internet users so that we cannot presume that 'average' voters submit their ballots anonymously. And even for those, probably few, voters who would use anonymous communication tools, their joint anonymity set may be too small to guarantee a significant level of everlasting privacy. We are aware of only one possible way to mitigate this issue: if ballots are published only at the end of the submission phase, but not before, then metadata alone does not help to link individual voters to their ballots.[5] However, this mitigation violates the *vote-and-go* paradigm of modern e-voting, which ensures that voters can verify instantly whether their ballots have been recorded as submitted and then leave the virtual voting booth.

Summary. To put it bluntly, everlasting privacy is approached mainly *constructively* in B-ID but only *hypothetically* in B-ANON. Hence, we conclude that the approach followed in B-ID is superior to the one in B-ANON.

4 B-ANON-A

We study existing protocols [2, 20, 27, 33, 40, 42, 62, 70] in the subclass B-ANON-A, where the election authority generates the voters' credentials (see Sec. 3). We identify three different groups which differ in whether and how eligibility of the anonymous ballots is ensured: no identities, blind signatures, and pseudonyms. In what follows, we elaborate on these groups and evaluate their properties.

4.1 No identities

The most trivial way to achieve everlasting privacy is to specify that ballots contain no information whatsoever about the respective voter's identity. Arapinis, Cortier, Kremer, and Ryan [2] describe and analyze such a version of Helios [1].[6]

4.1.1 Concept. We first describe the original Helios protocol [1] and then its variant without public voter identities.

In the setup phase of Helios, each voter V_i obtains an individual password from the election authority EA. In the submission phase of Helios, voter V_i encrypts her vote v_i under the trustee's public key pk, computes a NIZKP π_i to prove that her ciphertext $e_i \leftarrow \mathsf{E}(pk, v_i)$ is valid, uses her password to authenticate to the public bulletin board PBB, and then sends the tuple $(\mathsf{V}_i, e_i, \pi_i)$ to PBB. The bulletin board publishes $(\mathsf{V}_i, e_i, \pi_i)$ if the password belongs to V_i, if the NIZKP π_i is valid, and if π_i is not included in any other ballot published before. The encrypted votes are then either tallied homomorphically or with a mix net.

Obviously, the original Helios protocol does not provide (practical) everlasting privacy. The reason is that voters' ciphertexts are linked with their individual identities. Therefore, a computationally unbounded adversary, breaking secrecy of the ElGamal PKE scheme [28] employed in Helios, learns how all voters voted. Arapinis *et al.* [2] describe a version of Helios, which they call *Helios without identities*, where ballots are published in the form of (e_i, π_i), without any information about V_i's identity. In this way, even if an adversary is computationally unbounded after the election, vote privacy is supposed to be preserved.

4.1.2 Evaluation. Arapinis *et al.* [2] claim that Helios without identities provides practical everlasting privacy. To formally prove this statement, they propose and apply a definition of practical everlasting privacy to analyze Helios without identities with two different verification tools, AKISS [12] and ProVerif [6]. We did not find any issues that would contradict their privacy result. However, we

[5] We thank the anonymous reviewer who pointed this option out to us.
[6] Arapinis *et al.* [2] claim that this version was used in a student election at a Belgian university but they do not provide any reference to support this statement.

Table 3: Overview on the classification of e-voting protocols with everlasting privacy. Security properties not relying on any trust assumption are marked +. Revised trust assumptions are marked !. Trust assumptions are denoted as $\mathcal{T}$ for a full trust, $\mathcal{DT}$ for a distributed trust; trust to specific parties is denoted in subscript. Additional security properties are marked as CR for coercion-resistance, RF for receipt-freeness, ACC for accountability, followed by trust assumptions in parenthesis. *AS* denotes necessity of anonymous submission channel for privacy.

Sub-classes	Protocols	Distinguishing feature (∗) or used crypto primitive (†)	Public Verifiability	Everlasting Privacy	Additional Properties	Practical Limitations
B-ANON-A (Sec. 4)	[2]	∗ No identities	$\mathcal{T}_{\mathsf{EA}}$	$\mathcal{T}_{\mathsf{EA}}$		
	[27]	† Blind signatures	$\mathcal{T}_{\mathsf{EA}}$	+		*AS*
	[42]		$\mathcal{DT}_{\mathsf{EA}}$	+		Flawed crypto, *AS*
	[70]		$\mathcal{T}_{\mathsf{EA}}$	+		All voters online, not robust
	[33]		$\mathcal{T}_{\mathsf{EA}}$!	$\mathcal{T}_{\mathsf{EA}}$!	CR($\mathcal{T}_{\mathsf{EA}}$)	Small-scale elections, *AS*
	[20]	∗ Pseudonyms	$\mathcal{T}_{\mathsf{EA}}$	$\mathcal{DT}_{\mathsf{EA}}$		
	[40]		$\mathcal{T}_{\mathsf{EA}}$	$\mathcal{T}_{\mathsf{EA}}$	CR($\mathcal{T}_{\mathsf{EA}}$)	Small-scale elections
	[62]		$\mathcal{T}_{\mathsf{EA}}$	$\mathcal{T}_{\mathsf{EA}}$!	CR($\mathcal{T}_{\mathsf{EA}}$)!	
B-ANON-V (Sec. 5)	[35]	† Linkable ring signatures	+	+		Small-scale elections
	[45, 51–53]	† NIZKP	+	+	RF(+) [52], CR(+) [53]	Not robust [52, 53]
B-ID-HOM (Sec. 6)	[26, 69]	∗ Single tallier	+	$\mathcal{T}$		
	[22]	† Threshold decryption	+	$\mathcal{T}$	ACC(+)	
	[21, 29]	† Secret-sharing	$\mathcal{T}_{\mathsf{T}}$ [21], + [29]	$\mathcal{DT}$		
B-ID-MIX (Sec. 7)	[55, 56]	∗ Limited $\mathcal{DT}$ for Comp. P.	+	$\mathcal{T}$		On-site voting
	[22, 30, 34, 61]	∗ Arbitrary $\mathcal{DT}$ for Comp. P.	+	$\mathcal{T}$	ACC(+)	
	[11, 24–26]	∗ Arbitrary $\mathcal{DT}$ for Everl. P.	+	$\mathcal{T}$!		

note that Helios without identities is obviously not publicly verifiable because a corrupted EA can undetectably manipulate the election result by stuffing PBB with ballots of abstaining or even non-existing voters.

4.2 Blind signatures

Originally, the idea of using blind signatures for secure e-voting goes back to Fujioka, Okamoto, and Ohta [27], whose work is commonly referred to as the *FOO protocol*. Van de Graaf [70], who was the first one to observe *FOO* potentially offers everlasting privacy, proposed a technique to realize unconditionally anonymous submission channels for *FOO*. The *FOO* protocol is the basis of two further protocols [33, 42], designed to make *FOO* coercion-resistant [33] or to mitigate trust on the election authority EA [42], respectively.

4.2.1 Concept. We distinguish between two versions of e-voting with blind signatures. The advanced version, unlike the basic one, employs blind signatures with a special feature that is supposed to offer coercion-resistance in addition.

Basic version. A *blind signature scheme* [14] is an interactive protocol between a signer and a user with the goal that the user obtains a signature from the signer on a message but so that the signer does not learn the message. To this end, the user "blinds" the message and the signer signs the blinded message. The user can then "unblind" the signer's blind signature to obtain a signature on the actual message.

In [27, 42, 70], blind signatures are essentially employed as described next. Each voter blinds her vote and sends the blinded vote to the election authority EA. If the voter is eligible to vote, EA blindly signs the vote and returns its blind signature to the voter. Afterwards, the voter unblinds EA's blind signature and anonymously posts her signed vote on the public bulletin board PBB. Votes with valid signatures by EA are then tallied in clear.

Advanced version. A *conditional blind signature scheme* [33] is a blind signature scheme where the signer (EA) uses a secret bit in her blind signature to indicate whether the resulting signature on the actual message (vote) is valid or not. The user (voter) can check validity of the blind signature, but not of the unblinded one. Grontas, Pagourtzis, Zacharakis, and Zhang [33] employ conditional blind signatures as follows to achieve coercion-resistance.

In the setup phase of the election, EA creates a secret credential σ_i for each voter V_i and sends it to V_i. In the voting phase, if V_i is not under coercion, she sends her secret credential $\sigma = \sigma_i$ to EA when she requests a blind signature on her vote; otherwise, if V_i is under coercion, she uses any other ('fake credential') $\sigma = \sigma'$ for that purpose. Then, EA returns a blind signature to V_i in which EA secretly encodes that the unblinded signature is valid if and only if the submitted credential σ matches σ_i. Afterwards, in the tallying phase, ballots with invalid signatures are removed without revealing the links to individual voters. To this end, Grontas *et al.* employ the technique by Juels, Catalano, and Jakobsson [41], commonly referred to as the $\mathcal{JCJ}$ protocol.

4.2.2 Evaluation. We observe that neither the basic nor the advanced version provide secure solutions to e-voting with everlasting privacy under realistic assumptions.

Basic version. A significant problem of the original *FOO* protocol [27] is the fact that EA is able to issue blind signatures for all voters, even those ones who do not send such a request. Therefore, if EA is corrupted, it can secretly stuff the bulletin board with ballots of its own choice for all abstaining voters.. This problem motivated Kaim, Canard, Roux-Langlois, and Traoré [42] to distribute the role of EA by using the threshold blind signature scheme from [7]. However, that scheme [7] was shown to be flawed [36].

Van de Graaf [70] was the first one who observed the following potential of *FOO*: because the signature scheme [14] originally employed in *FOO* is unconditionally blinding, everlasting privacy can be achieved if the submission channels are unconditionally anonymous. Now, in order to realize such channels, [70] proposes a variant of Chaum's *Dining Cryptographers network (DC-net)* [15]. The author of [70] suggests to let all voters in the *FOO* protocol run this DC-net collectively. In this way, he argues, *FOO* offers everlasting privacy because voters can submit their ballots unconditionally anonymously. Creditably, [70] is the only work which addressed the problem of realizing unconditionally anonymous submission channels, but its construction illustrates our general criticism of B-ANON (see Sec. 3.3): the solution is neither realistic for most real-world elections nor robust because *all* voters would have to participate in the DC-net protocol.

Another issue of *FOO* follows from the fact that privacy in *FOO* solely reduces to the assumption that the submission channels are perfectly anonymous. In particular, unlike all other protocols in B-ANON-A, *FOO* does not additionally safeguard privacy by a computationally privacy-preserving tallying method.

Advanced version. We detect that, unlike originally claimed, the election authority EA in [33] needs to be trusted for *all* security properties (public verifiability, everlasting privacy, and coercion-resistance). Therefore, [33] is no more secure than a trivial e-voting protocol with a single completely trusted authority, which contradicts the basic idea of secure e-voting (see Sec. 2).

More precisely, the reason is that, according to the formal specification of [33], EA creates and thus knows the secret credentials of all voters. Hence, EA can impersonate any voter and is thus able to submit votes on all voters' behalf. Such an attack is impossible to detect in [33] because the authors specify that certain parties ("pro democratic organizations") submit (invalid) dummy votes for all voters to protect against coercers who want voters to abstain from voting. Moreover, for the same reason, EA also needs to be trusted for privacy and coercion-resistance. To see this, assume that EA is corrupted and secretly overwrites the votes of all voters except one, say V_i. Since the final election result then only consists of EA's votes and the one by V_i, EA learns how V_i voted, which breaks both privacy and coercion-resistance.

Furthermore, from the underlying $\mathcal{JCJ}$ protocol [41], the protocol by Grontas *et al.* [33] inherits the computational complexity of $\mathcal{JCJ}$'s tallying phase, which is quadratic in the number of voters. Hence, [33] cannot be deployed for large-scale elections. Grontas *et al.* acknowledge this issue and refer to [71] for mitigating it, but they do not provide any details.

4.3 Pseudonyms

Pseudonyms are used in [20, 40, 62] to combine public verifiability with practical everlasting privacy, and in case of [40, 62] with coercion-resistance in addition.

4.3.1 Concept. Using pseudonyms is a "quick and dirty way to obtain everlasting privacy" [40]. The election authority EA simply assigns to each voter V_i a (pseudo-)random number which serves as V_i's pseudonym. Under the assumption that EA keeps the individual links between voters and their pseudonyms secret, everlasting privacy follows.

In what follows, we describe how existing protocols in this group relate. We distinguish between Belenios [20] and [40, 62] since the latter ones were designed to offer coercion-resistance in addition; as we shall see, this difference has a significant impact on the necessary trust assumptions.

Belenios [20] essentially augments Helios (see above) with a public-key infrastructure among the voters in order to mitigate trust on the public bulletin board PBB. One of the election authorities, called the *Registrar* in Belenios, creates a verification/signing key pair (vk_i, ssk_i) for each voter V_i and sends the secret keys to the voters. The verification keys replace the voters' identities and thus serve as the voters' pseudonyms. In the submission phase, V_i signs her ballot with ssk_i and then uses her password to authenticate to PBB. Only ballots with valid signatures are tallied. Since the Registrar keeps the individual links between voters and their pseudonyms secret, practical everlasting privacy is supposed to follow.

The other two protocols [40, 62], which employ pseudonyms for everlasting privacy, aim for coercion-resistance in addition. To this end, they employ certain privacy-preserving techniques (which are not relevant for our purposes) that enable voters to secretly overwrite their possibly coerced votes. In both protocols, the election authority EA creates the secret credentials that voters use to create their anonymous ballots.[7]

4.3.2 Evaluation. We explain that all protocols [20, 40, 62] in this group need to trust EA for verifiability, at least to some degree, and for everlasting privacy. Moreover, [40, 62] even need to trust EA for all other security properties as well.

Cortier, Gaudry, and Glondu [20] claim that Belenios is verifiable if the election authority EA *or* the public bulletin board PBB are honest.[8] However, as we recall next, this claim is not correct. Hirschi, Schmid, and Basin [37] demonstrate that PBB in Belenios needs to be trusted for verifiability (and to a limited degree for privacy). Moreover, Baloglu, Bursuc, Mauw, and Pang [3] present different attacks against verifiability of Belenios if EA or PBB, but not necessarily both of them, are corrupted. Notably, in Sec. IV-C of [3], Baloglu *et al.* observe that the reason for these issues is the fact that Belenios uses pseudonyms to achieve everlasting privacy. In a different work, Baloglu *et al.* [4] show how to mitigate most, but not all of Belenios' verifiability issues they presented in [3].

We observe that in both protocols [40, 62], the election authority EA needs to be trusted for *all* security properties, i.e., public

[7] More precisely, the election authority responsible for assigning secret credentials is called *Certificate Authority* in [62] and *Registration Teller* in [40].

[8] In Belenios, the election authority who creates and distributes the verification/signing key pairs is called *Registrar* and the party who hosts PBB is called *Voting Server.*

verifiability, (everlasting) privacy, and coercion-resistance. As already mentioned in the evaluation of [33] (see Sec. 4.2), such an assumption conflicts with the basic idea of secure e-voting (see Sec. 2). We note that this limitation is acknowledged in [40] but not in [62]; in particular, our observation disproves the privacy and coercion-resistance theorems stated in [62].

The reason why EA needs to be trusted for all security properties follows from the fact that EA knows the links between all voters and their pseudonyms as well as all voters' secret credentials. A corrupted EA can exploit this knowledge to secretly overwrite any voter's choice by anonymously submitting a new ballot on that voter's behalf. Therefore, EA needs to be trusted for public verifiability. Moreover, analogously to [33] (see Sec. 4.2), a corrupted EA can target a particular voter V_i and impersonate all voters except V_i so that the final result consists only of EA's votes and V_i's vote; hence, EA also needs to be trusted for (everlasting) privacy and coercion-resistance.

We also note that, while [62] can handle large-scale elections efficiently, the computational complexity of the tallying phase in [40] is quadratic in the number of voters, which it inherits from the underlying JCJ protocol [41]. Hence, [40] cannot be deployed efficiently for large-scale elections.

4.4 Summary

We demonstrated that there does not exist a completely satisfying solution for secure e-voting with everlasting privacy in B-ANON-A, even if unconditionally anonymous channels are in place. The main reason is that all existing protocols in B-ANON-A put much trust in the election authority EA. Among all protocols in this sub-class, the impact of a corrupted EA seems most limited in Belenios [20], which therefore offers the most reasonable solution in B-ANON-A. However, it remains an open problem to precisely analyze to which degree EA needs to be trusted in Belenios, or, even better, to completely resolve this issue.

5 B-ANON-V

We study existing protocols [35, 45, 51–53] in the sub-class B-ANON-V, where the voters themselves generate their credentials (see Sec. 3). We identify two different groups which differ in how eligibility of the anonymous ballots is ensured: linkable ring signatures and membership ZKP. In what follows, we elaborate on these groups and evaluate their properties.

5.1 Linkable ring signatures

Linkable ring signatures are the key technique of the *VOTOR* protocol by Haines and Boyen [35].

5.1.1 Concept. *Ring signatures* [63] enable users (voters) of a known group (set of eligible voters) to sign messages anonymously. Unlike blind signatures, that are used in the *FOO*-like protocols in B-ANON-A (see Sec. 4.2), ring signatures do not require a central signing party.

In *VOTOR*, ring signatures are employed as follows. Each voter V_i generates a public/private key pair (vk_i, ssk_i) for the ring signature scheme. The verification key vk_i is used to update the voters' joint verification key vk; the secret signing key ssk_i remains private. When the voter creates her ballot, she signs her vote using ssk_i. Then, the voter submits her signed vote to the public bulletin board PBB via the onion router TOR. At the end of the voting phase, it is verified for each public vote whether the corresponding signature is valid w.r.t. vk. All votes which pass this test are then tallied.

The ring signature scheme employed in *VOTOR* offers two additional properties: linkability and forward-security. *Linkability* prevents voters from casting multiple votes. *Forward-security* allows voters to update their secret keys ssk_i in each election without changing their public verification keys vk_i so that even if updated private keys are revealed, then this does not leak any information about previously used private keys. For this purpose, *VOTOR* employs the ring signature scheme proposed in [8], which is based on the existence of bilinear or multilinear maps.

5.1.2 Evaluation. The authors of [35] state that *VOTOR* offers public verifiability as well as practical everlasting privacy. We did not discover any issues that would contradict their statement. However, the amount of work required to verify a signature in the employed scheme is linear in the size of the electorate; this makes the total computation quadratic in the size of the electorate which is infeasible for larger-scale elections.

5.2 Membership ZKP

Membership ZKPs are the key technique of the e-voting protocol by Locher and Haenni [51]. This protocol was extended in [52, 53] to offer receipt-freeness [52] or coercion-resistance [53], respectively; these protocols use the same membership ZKP as [51] and differ from [51] mainly in the tallying phase. Furthermore, [51] was implemented in [45].

5.2.1 Concept. The purpose of a *membership ZKP* is to enable a prover (voter) to prove knowledge of a secret key ssk_i that belongs to one of the public keys stored in some list $\vec{vk}$ (all voters' verification keys), without revealing to which one specifically. Locher and Haenni propose a specific membership NIZKP [51] which additionally allows for identifying proofs that used the same ssk_i.

This membership NIZKP is employed in [45, 51–53] as follows. In the setup phase, each voter V_i creates a public/private key pair (vk_i, ssk_i). The public key vk_i is added to the list $\vec{vk}$, while the secret key ssk_i remains private. When the voter submits a vote, she uses her secret credentials ssk_i to create a membership NIZKP π_i for $\vec{vk}$. She then appends π_i to her vote v_i and anonymously submits the resulting pair (v_i, π_i) to the public bulletin board PBB. At the end of the voting phase, votes with invalid NIZKPs are removed. After that, the respective last votes that use the same ssk_i for their membership NIZKPs are identified and then tallied.

5.2.2 Evaluation. Locher and Haenni state that their protocol offers public verifiability as well as practical everlasting privacy [51]. We did not discover any issues that would contradict their statement. Moreover, the authors implemented their membership ZKP and provided detailed benchmarks to demonstrate that their protocol can be used for larger-scale elections.

We note, however, that the fact that the implementation [45] of [51] abstracts away from the anonymous channels exemplifies our general criticism of B-ANON (see Sec. 3.3). This limitation is particularly problematic in [51] because, similarly to *FOO* (see Sec. 4),

privacy solely reduces to the existence of anonymous submission channels.

We observe that both extensions [52, 53] of [51] share the same robustness issue. In order to achieve receipt-freeness or coercion-resistance, respectively, voters can submit an arbitrary number of ballots. Now, the problem is that the complexity of the tallying phase grows linearly [52] or even quadratically [53] in the number of submitted ballots. Therefore, if a *single* corrupted voter submits many ballots, the tallying phase becomes inefficient. Due to the anonymous ballot submission, such a corrupted voter cannot be identified. As a result, both protocols [52, 53] are not robust because they can only guarantee that the final result is delivered under the unrealistic assumption that all voters are honest.

5.3 Summary

We demonstrated that [51] is the only known practically efficient solution for secure e-voting with everlasting privacy in B-ANON-V, although under the (arguable) assumption that all voters' submission channels are unconditionally anonymous (see Sec. 3.3). All other protocols are inefficient for larger-scale elections [35] or not robust [52, 53].

6 B-ID-HOM

We study existing protocols [21, 22, 26, 29, 69] in the sub-class B-ID-HOM, where voters' identifiable ballots are homomorphically aggregated (see Sec. 3). We identify three different groups which differ in whether and how trust for privacy is distributed among the talliers: no distribution, distributed decryption, and secret-sharing. In what follows, we elaborate on second and the third group and evaluate their properties.

We do not elaborate on existing protocols [26, 69] in the first group, where trust is not distributed, because they can be regarded as special cases of the two approaches in which trust is actually distributed.

We remind the reader that homomorphic tallying allows for handling simple ballot types, whereas mixing ballots (see Sec. 7) also allows for processing complex ballots.

6.1 Distributed decryption

In the homomorphic version of *Perfectly Private Audit Trail (PPAT)* by Cuvelier, Pereira, and Peters [22], trust is distributed via threshold decryption.

6.1.1 Concept. In the voting phase, each voter encrypts her vote v_i under the trustees' joint public key pk as $e_i \leftarrow \mathsf{E}(pk, v_i)$. Then, instead of posting e_i on the public bulletin board PBB (which would break everlasting privacy), the voter sends e_i to a special party, called the *secret bulletin board* SBB. Unlike PBB, to which everyone can access, the content of SBB is only accessible by the trustees.

Now, the main cryptographic idea of *PPAT* is that the homomorphic PKE employed offers a special property, *Commitment-Consistent Encryption* (CCE) [22]. This feature enables everyone, given ciphertext $e = \mathsf{E}(pk, m)$, to efficiently and deterministically derive a commitment $c = \mathsf{C}(prm, m, r)$ to the encrypted message m. The opening values (m, r) of the commitment c can be computed from e with the secret key sk.

Using the CCE property, SBB derives a commitment $c_i = \mathsf{C}(prm, v_i, r_i)$ from each ciphertext $e_i = \mathsf{E}(pk, v_i)$ privately submitted by V_i, and posts c_i on PBB. In the tallying phase, each trustee T_j uses its share sk^j of the secret key sk to compute the partial opening values (v^j, r^j) of the aggregated commitments $c \leftarrow \sum_i c_i$ from the aggregated ciphertexts $e \leftarrow \sum_i e_i$. Afterwards, T_j posts (v^j, r^j) on SBB. These partial opening values are then combined to obtain the full opening values (v, r). Eventually, (v, r) is published on PBB, where v determines the final election result. The correctness of v can be verified by checking $\mathsf{O}(prm, c, v, r) \stackrel{?}{=} 1$.

6.1.2 Evaluation. Cuvelier *et al.* stated their protocol [22] guarantees public verifiability and practical everlasting privacy. Cuvelier *et al.* discuss that their protocol provides two advantages over the secret-sharing approach (see below). Firstly, they argue that their protocol also allows for resolving possible disputes in the submission phase and thus, unlike [21], potentially offers accountability (see Sec. 2). Secondly, they observe that the voters' work load is independent of the number of trustees, whereas it increases linearly with secret-sharing. We agree with Cuvelier *et al.* in both points, but we think that the second advantage is practically negligible because in real-world elections at most a handful of trustees are typically employed.

Cuvelier *et al.* propose different instantiations of generic CCE schemes. One of them is an extended version of ElGamal PKE [28], where the derived commitments are (unconditionally hiding) Pedersen commitments [60]; this instantiation is practically efficient.

We conjecture that the homomorphic version of [22] achieves the security properties as stated by Cuvelier *et al.* [22]. Since Cuvelier *et al.* demonstrated that their abstract primitives can be instantiated efficiently, we conclude that their approach offers a reasonable solution for secure e-voting with everlasting privacy.

6.2 Secret-sharing

The idea of using secret-sharing for distributing trust among the talliers goes back to Cramer, Franklin, Schoenmakers, and Yung [21]. Ge, Chau, Gonsalves *et al.* [29] present a full-fledged back-end of an e-voting *system* based on Cramer *et al.*'s protocol [21]; in particular, Ge *et al.* address and resolve a critical issue from which Cramer *et al.* abstracted away from originally (see below).

6.2.1 Concept. We explain how [21] works on a high level. For the sake of simplicity, we make two assumptions. First, voters can choose between two possible candidates, encoded as 0 and 1. In order to extend the protocol to n candidates, the protocol can essentially be run in parallel n times. Second, we use full-threshold additive secret-sharing instead of arbitrary threshold secret-sharing.

The following cryptographic primitives are employed in [21]. First, a commitment scheme $C = (\mathsf{KG}_c, \mathsf{C}, \mathsf{O})$, which is unconditionally hiding, computationally binding, and additively homomorphic. Second, a NIZKP of knowledge Π for the relation $\mathcal{R} = \{((prm, c), (m, r)) \colon c = \mathsf{C}(prm, m, r) \wedge m \in \{0, 1\}\}$, which states that commitment c commits to one of the two possible candidates.

In the *setup phase*, the election authority EA generates parameters prm and publishes prm on PBB.

In the *voting phase*, each voter V_i first secretly shares her vote v_i among the talliers $\mathsf{T}_1, \ldots, \mathsf{T}_m$ as $v_i = \sum_j v_i^j$. Then, for each share

v_i^j, voter V_i computes a commitment $c_i^j \leftarrow \mathsf{C}(prm, v_i^j, r_i^j)$. After that, V_i computes a NIZKP π_i for statement (prm, c_i) with witness (v_i, r_i), where $c_i \leftarrow \prod_j c_i^j$ and $r_i \leftarrow \sum_j r_i^j$. Eventually, V_i publishes her ballot $b_i \leftarrow (i, (c_i^j)_j, \pi_i)$ on PBB and privately sends all opening values (v_i^j, r_i^j) to the respective trustees T_j. If (v_i^j, r_i^j) is not a valid opening for c_i^j published on PBB, then T_j posts a complaint on PBB.

In the *tallying phase*, all ballots b_i, for which the NIZKP π_i is invalid or for which a complaint by a trustee was filed, are discarded. After that, each trustee T_j aggregates its obtained opening values (of the remaining ballots) as $v^j \leftarrow \sum_i v_i^j$ and $r^j \leftarrow \sum_i r_i^j$, and publishes (v^j, r^j). The correctness of T_j's output is verified as $\mathsf{O}(prm, c^j, v^j, r^j) \stackrel{?}{=} 1$, where $c^j \leftarrow \prod_j c_i^j$. The final election result $v \leftarrow \sum_j v^j$ is accepted if and only if all of the previous checks are positive.

6.2.2 Evaluation. Let us first explain why the original secret-sharing protocol by Cramer *et al.* [21], as described before, does not offer public verifiability.

Since the commitment scheme is homomorphic and binding, the final result v is accepted (with overwhelming probability) if and only if $v = \sum_i v_i$, where v_i is the message in V_i's aggregated commitments c_i. Hence, the talliers cannot cheat during tallying. Moreover, due to the soundness of the voters' NIZKP π, the voters can only commit to valid choices.

On first sight, these two properties may seem sufficient for public verifiability, but there exists a significant gap. To see this, assume that trustee T_j claims that it cannot process V_i's commitment c_i^j because V_i did not submit valid opening values for c_i^j; consequently, V_i's ballot is discarded. Now, the problem is that, without further means, it is impossible to distinguish in such a situation whether T_j is telling the truth or not. In particular, if T_j is corrupted, it can make this claim falsely so that V_i's ballot is wrongly excluded from the tallying. Hence, public verifiability is not guaranteed in [21].

Fortunately, as we will explain in what follows, the public verifiability issue of [21] can be (and has been) resolved in different ways.

Solution: receipts. In order to close the verifiability gap, Ge *et al.* [29] specify that voter V_i sends the tuple (c_i^j, v_i^j, r_i^j) to trustee T_j and that the trustee returns a signature s_i^j on the commitment c_i^j if (v_i^j, r_i^j) is a valid opening for c_i^j. The voter then publishes on PBB her commitments $(c_i^j)_j$ together with the trustees' signatures $(s_i^j)_j$, which serve as receipts. Of course, a dishonest tallier T_j could still refuse to reply at all to V_i's request, but that problem can be mitigated by practical means (e.g., auditors can send test requests to the talliers). We therefore find that Ge *et al.*'s modification of [21] closes the gap described above and hence offers public verifiability under realistic assumptions.

Solution: secret bulletin board. We note that, as an alternative solution, we can employ a secret bulletin board SBB to which only trustees can access (like in [22]), as described next. Voters post on SBB their commitments $(c_i^j)_j$ (and NIZKP π_i) together with their opening values, all encrypted under the respective talliers' public keys, i.e., $e_i^j \leftarrow \mathsf{E}(pk_j, (v_i^j, r_i^j))$. The secret bulletin board SBB uses a cryptographic hash function to hash all ciphertexts $(e_i^j)_j$ to obtain $(h_i^j)_j$, and then posts the ballot $(i, (c_i^j)_j, \pi_i, (h_i^j)_j)$ on the public bulletin board PBB. The voter can verify whether her ballot was published correctly. Now, using a PKE scheme that allows for verifiable decryption, we specify that a tallier needs to verifiably decrypt a ciphertext e_i^j on PBB whenever it claims that e_i^j does not contain valid opening values for c_i^j; the hash h_i^j ensures that the publicly decrypted ciphertext is in fact the one privately submitted by V_i. In this way, a corrupted tallier can no longer make the above claim falsely.

Comparison of solutions. In Figure 1 (App. B), we present the voting phase in [21] and the two solutions. Both solutions, the one by Ge *et al.* [29] and the alternative one, provide their own balances between public verifiability and everlasting privacy. In terms of verifiability, the alternative solution we propose is superior to the one in [29] because it not only mitigates but completely removes all trust on the talliers for verifiability. For the same reason, the alternative approach potentially offers accountability, i.e., individual identification of all misbehaving parties (recall Sec. 2). However, unlike [29], the alternative approach has the drawback that all talliers need to be trusted for everlasting privacy because they learn the encrypted opening values on SBB.

Altogether, we conjecture that, with one of the improvements in place, the secret-sharing approach offers public verifiability (and optionally even accountability) as well as everlasting privacy towards the public (and optionally even towards less than a threshold of the trustees).

Finally, we note that the secret-sharing approach can be deployed efficiently for large-scale elections with simple ballot types, as demonstrated by Ge *et al.* [29].

6.3 Summary

For simple ballot types, secret-sharing and distributed decryption are both reasonable approaches for secure e-voting with everlasting privacy. The secret-sharing approach has, however, some advantageous features. First, unlike the distributed decryption approach, which employs specific cryptographic primitives (CCE), the secret-sharing approach can be instantiated with a larger class of primitives; this feature may be helpful to offer receipt-freeness, which is still an open problem. Second, depending on how the verifiability gap is closed, the secret-sharing approach provides everlasting privacy even if a future adversary gains private data of some trustees.

The most pressing open problem is to formally analyze on the protocol level our conjectures on the security of the secret-sharing and distributed decryption approaches.

7 B-ID-MIX

We study existing protocols [11, 22, 25, 26, 30, 34, 55, 56, 61] in the sub-class B-ID-MIX, where voters' identifiable ballots are mixed (see Sec. 3). We identify three different groups which differ in whether and how trust is distributed for privacy: limited distribution for computational privacy, arbitrary distribution for computational privacy, and arbitrary distribution for everlasting privacy. In what follows, we elaborate on these groups and evaluate their properties.

We remind the reader that the main advantage of mixing ballots over homomorphically aggregating (see Sec. 6) is the property that arbitary (possibly very complex) ballots can be processed efficiently.

7.1 Limited distribution of trust for computational privacy

Originally, the idea of providing everlasting privacy for verifiable e-voting via mixing/shuffling ballots goes back to Moran and Naor [55]. They propose a verifiable on-site (i.e., booth) e-voting protocol which employs an interactive zero-knowledge proof (ZKP) protocol for shuffling unconditionally hiding commitments. Because that ZKP is specifically tailored to shuffle and open a vector of commitments in a single step, the overall e-voting protocol [55] is centered around a single tallier who learns how each individual voter votes. In order to mitigate trust on the tallier, Moran and Naor subsequently presented a new verifiable on-site e-voting protocol [56], called *Split-Ballot*, with two separate talliers.

7.1.1 Concept. On a high-level, *Split-Ballot* works as follows. In the setup phase, each trustee $\mathsf{T}_j \in \{\mathsf{T}_0, \mathsf{T}_1\}$ creates a mask t_i^j for each voter V_i, commits to the mask as $c_i^j \leftarrow \mathsf{C}(t_i^j)$, and publishes the vector of all unconditionally hiding commitments $(c_i^j)_i$ on PBB. When voter V_i wants to submit a vote, she obtains from each trustee T_j its mask t_i^j. The voter then uses t_i^0, t_i^1 to perfectly hide her vote v_i as $s_i \leftarrow v_i - t_i^0 - t_i^1$ and publishes (i, s_i) on PBB.

At the start of the tallying phase, for each voter V_i, $d_i \leftarrow \mathsf{C}(s_i) + c_i^0 + c_i^1$ is (publicly) computed, which is a commitment to V_i's vote v_i due to the homomorphic property of the commitment scheme. Afterwards, the commitment vector $(d_i)_i$ is first shuffled (i.e., privately re-encrypted and permuted) by T_0 and then by T_1; we denote the resulting commitment vector by $(\tilde{d}_i)_i$. Then, T_0 and T_1 use their knowledge of the opening values in $(c_i^0)_i$ and $(c_i^1)_i$, respectively, to jointly open $(\tilde{d}_i)_i$ and to create a proof π that the resulting plaintext vector $(\tilde{v}_i)_i$ is in fact the result of shuffling and opening $(d_i)_i$; the proof π is unconditional ZK and neither of trustees T_j learns the masks of the respective other trustee T_{1-j}. The outcome vector $(\tilde{v}_i)_i$, which consists of all voters' choices in a secretly permuted order, denotes the final election result.

7.1.2 Evaluation. Moran and Naor formally proved in the Universal Composability (UC) framework that *Split-Ballot* guarantees practical everlasting privacy, computational privacy if one trustee is honest, and public verifiability. Furthermore, Arapinis, Cortier, Kremer, and Ryan [2] analyzed everlasting privacy of a simplified version of *Split-Ballot* in ProVerif.

Despite its thorough security analyses, *Split-Ballot* has two significant disadvantages. Firstly, the protocol was specifically designed (even on the cryptographic level) for elections with exactly two trustees. Secondly, the protocol can only be employed for on-site (booth) voting, but not for remote (Internet) voting.

7.2 Arbitrary distribution of trust for computational privacy

The mix net version of Cuvelier *et al.*'s *perfectly private audit trail (PPAT)* framework [22] (recall Sec. 6) is designed to distribute trust for computational privacy among an arbitrary number of talliers.

7.2.1 Concept. The main idea of [22] is to shuffle *in parallel* unconditionally hiding commitments to the voters' choices on PBB and the corresponding encrypted opening values of the public commitments on SBB. On the cryptographic level, in order to connect the public trail on PBB and the (perfectly) private trail on SBB, again commitment-consistent encryption (CCE) is employed (recall Sec. 6).

More precisely, the protocol works as follows. The voting phase is the same as in the homomorphic version: voter V_i encrypts her vote v_i under the trustees' joint public key pk as $e_i \leftarrow \mathsf{E}(pk, v_i)$, posts e_i on SBB, SBB derives a commitment $c_i = \mathsf{C}(v_i, r_i)$ from each ciphertext e_i, and publishes c_i on PBB.

In the tallying phase, the vectors $\vec{e}_0 = (e_i)_i$ are processed by a set of mix servers $\mathsf{M}_1, \ldots, \mathsf{M}_l$ as follows. The first mix server takes as input $\vec{e}_0$, re-encrypts all ciphertexts in $\vec{e}_0$, and then permutes the re-encrypted ciphertexts to obtain a shuffled ciphertext vector $\vec{e}_1$. The first mix server M_1 posts $\vec{e}_1$ on SBB, which will be the input of the second mix server M_2, and so on. Using the CCE feature, a commitment vector $\vec{c}_k$ is derived from each shuffled ciphertext vector $\vec{e}_k$ and then published on PBB. Eventually, the trustees $\mathsf{T}_1, \ldots, \mathsf{T}_m$ use their secret key shares $sk_1, \ldots, sk_m$ to jointly compute an opening $(\vec{v}, \vec{r})$ of the final commitment vector $\vec{c}_l$. The final election result $\vec{v}$ consists of the voters' individual votes, secretly shuffled according to the mix servers overall permutation. The correctness of $\vec{v}$ can be verified by checking $\mathsf{O}(\vec{c}_l, \vec{v}, \vec{r}) \stackrel{?}{=} 1$.

In order to ensure correctness of all intermediate shuffles, Cuvelier *et al.* [22] employ a NIZKP for proving that a ciphertext vector $\vec{e}_k$ is in fact a result of shuffling $\vec{e}_{k-1}$. This NIZKP is "CCE-compatible" in the following sense: from a proof π_k for the ciphertext vector pair $(\vec{e}_{k-1}, \vec{e}_k)$, a proof π'_k can be derived for the corresponding commitment vector pair $(\vec{c}_{k-1}, \vec{c}_k)$. Now, Cuvelier *et al.* specify that each mix server M_k posts such a proof π_k on SBB; the derived proof π'_k is published on PBB. Before the trustees open the final commitment vector, they need to verify correctness of all these NIZKPs. We note that these checks are also necessary for privacy in a re-encryption mix net.[9]

7.2.2 Evaluation. We conjecture that the mixing version of [22] achieves all security properties it was designed for originally: practical everlasting privacy, computational privacy under the assumption at least one mix server and at least a threshold of trustees are honest, public verifiability, and accountability. Because Gjøsteen, Haines, and Solberg [30] demonstrate that the *PPAT* protocol can be instantiated to achieve a performance similar to state-of-the-art NIZKP of shuffle [68], we conclude that the *PPAT* protocol offers a reasonable solution for secure e-voting with everlasting privacy that can even handle complex ballots.

We mention two works built upon [22]. Pereira and Rønne [61] show how to realize a special voting method, called *quadratic voting* [50], in the *PPAT* setting. Haines [34] proposes a cast-as-intended protocol for *PPAT* to mitigate trust on the voting devices. We studied these extensions and confirm that they achieve the additional features they were designed for.

[9] Without such checks, if a mix server M_k is corrupted, it can manipulate its outcome $\vec{e}_k$ so that the final result leaks how individual voters voted. For example, M_1 could replace all ciphertexts except for the one by some voter V_i. Then, the final result would consist only of M_1's votes and V_i's vote. In this way, M_1 learns how V_i voted.

7.3 Arbitrary distribution of trust for everlasting privacy

About at the same time when Cuvelier *et al.* introduced the concept of *PPAT* [22], in a parallel line of research, Buchmann, Demirel, and van de Graaf [11] published a similar approach to *PPAT* but with a more ambitious goal: unlike in [22], trust among mix servers should not only be distributed for computational but also for everlasting privacy. In her PhD thesis, Demirel [24], proved this protocol secure in a cryptographic framework, and Arapinis *et al.* [2] proved everlasting privacy of this protocol in a symbolic model, using automated verification tools.

Buchmann *et al.*'s protocol [11] is employed in [25, 26] to provide unconditional privacy in the mix-net version of Helios [1] and in Prêt à Voter [65], respectively.

7.3.1 Concept. The following cryptographic primitives are used in [11]. First, a commitment scheme $\mathcal{C} = (\mathsf{KG}_c, \mathsf{C}, \mathsf{O})$, which is unconditionally hiding, computationally binding, and homomorphic. Second, a PKE scheme $\mathcal{E} = (\mathsf{KG}_e, \mathsf{E}, \mathsf{D})$, which is IND-CPA-secure[10] and homomorphic. Third, a NIZKP of "correct re-encryption" to prove that "the set of output values is a valid shuffle of the set of input values" [11], and a NIZKP of "consistency" to "privately prove that the same permutation and random values have been used to rerandomize the published commitments and to reencrypt the corresponding private encrypted opening values" [11]; the exact relations are not defined in [11] and hence remain unclear.

In the *voting phase*, each voter V_i commits to her vote v_i as $c_i \leftarrow \mathsf{C}(prm, v_i, r_i)$. Then, V_i encrypts the opening values of c_i separately as $e_i^0 \leftarrow \mathsf{E}(pk, v_i)$ and $e_i^1 \leftarrow \mathsf{E}(pk, r_i)$, under the trustees' public key pk. Afterwards, the voter creates a proof of consistency π_i for (c_i, e_i^0, e_i^1) and submits $b_i \leftarrow (c_i, e_i^0, e_i^1, \pi_i)$ to the first mix server M_1. At the end of the voting phase, M_1 publishes the commitments c_i of the ballots b_i it received on the bulletin board PBB as a vector C_0 and keeps the corresponding ciphertext vectors E_0^0 and E_0^1 secret.

In the *mixing phase*, starting with the first mix server M_1, each mix server M_k takes as input $(C_{k-1}, E_{k-1}^0, E_{k-1}^1)$ that it either received from the voters (in case of M_1) or from the preceding mix server M_{k-1}. First, M_k rerandomizes each commitment $c_{k-1,i}$ from C_{k-1} using some fresh randomness $\rho_{k,i}$ and homomorphically adds $\rho_{k,i}$ to the associated encrypted randomness in $e_{k-1,i}^1$ from E_{k-1}^1. Furthermore, M_k rerandomizes the vector of encrypted votes E_{k-1}^0. Afterwards, M_k shuffles the resulting vectors $(\tilde{C}_k, \tilde{E}_k^0, \tilde{E}_k^1)$ into (C_k, E_k^0, E_k^1), all permuted according to the same random permutation. Then, M_k computes a NIZKP of "correct re-encryption" and posts it on PBB. Eventually, M_k sends (C_k, E_k^0, E_k^1) to M_{k+1} (or the trustees in case of the last mix server) via its private channel, together with a proof of "consistency".

In the *opening/decryption phase*, the trustees use their shares of the secret key sk to jointly compute $V^* \leftarrow \mathsf{D}(sk, E_m^0)$ and $R^* \leftarrow \mathsf{D}(sk, E_m^1)$, and publish (V^*, R^*) on the bulletin board PBB.

In the *verification phase*, it is checked whether (V^*, R^*) is a valid opening for C_m and whether all proofs of correct re-encryption published by the mix servers on PBB are valid.

[10] In fact, Buchmann *et al.* write that $\mathcal{E}$ is IND-CCA-secure, but that would be in conflict with its homomorphic property. We assume that they meant IND-CPA-secure.

7.3.2 Evaluation. The original protocol description, both in Buchmann *et al.*'s conference paper [11] and in Demirel's PhD thesis [24], is imprecise in several important aspects. For example, it is unclear how mix server M_{k+1} can verify M_k's proof of consistency when it does not know M_k's input. In what follows, we condone these issues and focus on the main conceptual shortcoming.

We discover that, in fact, trust is *not* distributed in [11], neither in terms of computational nor everlasting privacy. Our finding therefore disproves Buchmann *et al.*'s original privacy theorem [11] (and its proof in Sec. 5.2.2 of [24]), as well the everlasting privacy result in [2].[11] The reason for this issue is that, due to the design of [11], it cannot be guaranteed that the mix servers involved share the same view on the private trail of votes. For example, if the first mix server M_1 is corrupted, it can replace all ciphertexts except for the one by some voter V_i. Then, the final result consists only of M_1's votes and V_i's vote. In this way, M_1 learns how V_i voted. But also more advanced attacks to break privacy of several voters are feasible by exploiting the homomorphic property of the PKE scheme (see, e.g., [58]).

Since this issue stems from the design of [11], we conclude that there does not exist a secure e-voting protocol in the literature that can simultaneously handle arbitrary ballots and mitigate trust on the talliers for everlasting privacy.

7.4 Summary

The mixing version of *PPAT* [22] is the only known approach in B-ID-MIX in which trust for computational privacy is actually distributed among an arbitrary number of talliers. Both other approaches require trust in all (or all but one) talliers. The most pressing open problem is to formally analyze our conjectures on the security of the *PPAT* protocol.

8 CONCLUSION

We demonstrated that there exist four promising approaches [20–22, 51] among the numerous proposals for secure e-voting with everlasting privacy. These solutions offer the potential to guarantee everlasting privacy in real elections. We explained, however, that these approaches significantly differ in the assumptions that they need to make for everlasting privacy. While [20, 51] need to assume that voters submit their ballots anonymously, the other two approaches can avoid this, as we argued, often unrealistic assumption. Therefore, [21, 22] are preferable whenever distributing the trustee is feasible.

We identified two important open problems, one of theoretical and the other one of practical nature. First, it is fundamental to formally analyze the security of all promising protocols [20–22, 51]. Second, it is desirable to realize the two strongest proposals [21, 22] so that they can be deployed to guarantee everlasting privacy of elections in the real world, not only in theory.

[11] Although the everlasting privacy framework by Arapinis *et al.* [2] seems sound and the verification tools they used are established, the authors overlooked that [11] does not distribute trust among mix servers, probably because they modeled this protocol with a single mix server only.

ACKNOWLEDGMENTS

Thomas Haines is the recipient of an Australian Research Council Australian Discovery Early Career Award (project number DE220100595). Rafieh Mosaheb was supported by the Luxembourg National Research Fund (FNR), under the CORE project EquiVox (C19/IS/13643617/EquiVox/Ryan). 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).

REFERENCES

[1] Ben Adida. 2008. Helios: Web-based Open-Audit Voting. In *USENIX Security 2008*. 335–348.
[2] Myrto Arapinis, Véronique Cortier, Steve Kremer, and Mark Ryan. 2013. Practical Everlasting Privacy. In *POST 2013*. 21–40.
[3] Sevdenur Baloglu, Sergiu Bursuc, Sjouke Mauw, and Jun Pang. 2021. Election Verifiability Revisited: Automated Security Proofs and Attacks on Helios and Belenios. In *IEEE CSF 2021*. 1–15.
[4] Sevdenur Baloglu, Sergiu Bursuc, Sjouke Mauw, and Jun Pang. 2021. Provably Improving Election Verifiability in Belenios. In *E-Vote-ID 2021, Proceedings*. 1–16.
[5] David Bernhard, Véronique Cortier, David Galindo, Olivier Pereira, and Bogdan Warinschi. 2015. SoK: A Comprehensive Analysis of Game-Based Ballot Privacy Definitions. In *IEEE S&P 2015*. 499–516.
[6] Bruno Blanchet, Martín Abadi, and Cédric Fournet. 2008. Automated verification of selected equivalences for security protocols. *J. Log. Algebraic Methods Program.* 75, 1 (2008), 3–51.
[7] Samuel Bouaziz-Ermann, Sébastien Canard, Gautier Eberhart, Guillaume Kaim, Adeline Roux-Langlois, and Jacques Traoré. 2020. Lattice-based (Partially) Blind Signature without Restart. *IACR Cryptol. ePrint Arch.* (2020), 260.
[8] Xavier Boyen and Thomas Haines. 2018. Forward-Secure Linkable Ring Signatures. In *ACISP 2018*. 245–264.
[9] Xavier Boyen, Thomas Haines, and Johannes Müller. 2020. A Verifiable and Practical Lattice-Based Decryption Mix Net with External Auditing. In *ESORICS 2020*. 336–356.
[10] Xavier Boyen, Thomas Haines, and Johannes Müller. 2021. Epoque: Practical End-to-End Verifiable Post-Quantum-Secure E-Voting. In *IEEE EuroS&P 2021*. 272–291.
[11] Johannes Buchmann, Denise Demirel, and Jeroen van de Graaf. 2013. Towards a Publicly-Verifiable Mix-Net Providing Everlasting Privacy. In *FC 2013, Revised Selected Papers*. 197–204.
[12] Rohit Chadha, Ştefan Ciobâcă, and Steve Kremer. 2012. Automated Verification of Equivalence Properties of Cryptographic Protocols. In *ESOP 2012. Proceedings*, Helmut Seidl (Ed.), Vol. 7211. 108–127.
[13] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. 2016. BeleniosRF: A Non-interactive Receipt-Free Electronic Voting Scheme. In *ACM CCS 2016*. 1614–1625.
[14] David Chaum. 1983. Blind Signature System. In *CRYPTO 1983*. 153.
[15] David Chaum. 1988. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *J. Cryptol.* (1988), 65–75.
[16] Vincent Cheval, Charlie Jacomme, Steve Kremer, and Robert Künnemann. 2022. SAPIC+: Protocol Verifiers of the World, Unite!. In *USENIX Security Symposium (USENIX Security), 2022*.
[17] 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*. Springer, 191–199.
[18] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. 2008. Civitas: Toward a Secure Voting System. In *IEEE S&P 2008*. 354–368.
[19] Véronique Cortier, David Galindo, Ralf Küsters, Johannes Müller, and Tomasz Truderung. 2016. SoK: Verifiability Notions for E-Voting Protocols. In *IEEE S&P 2016*. 779–798.
[20] 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*. Springer, 214–238.
[21] Ronald Cramer, Matthew K. Franklin, Berry Schoenmakers, and Moti Yung. 1996. Multi-Autority Secret-Ballot Elections with Linear Work. In *EUROCRYPT 1996*. 72–83.
[22] Edouard Cuvelier, Olivier Pereira, and Thomas Peters. 2013. Election Verifiability or Ballot Privacy: Do We Need to Choose?. In *ESORICS 2013*. 481–498.
[23] Ivan Damgård, Daniel Escudero, and Divya Ravi. 2021. Information-Theoretically Secure MPC Against Mixed Dynamic Adversaries. In *TCC 2021*. 591–622.
[24] Denise Demirel. 2013. *Universally verifiable poll-site voting schemes providing everlasting privacy*. Ph. D. Dissertation. Darmstadt University of Technology.
[25] Denise Demirel, Maria Henning, Jeroen van de Graaf, Peter Y. A. Ryan, and Johannes Buchmann. 2013. Prêt à Voter Providing Everlasting Privacy. In *VoteID 2013*. 156–175.
[26] Denise Demirel, Jeroen van de Graaf, and Roberto Samarone dos Santos Araújo. 2012. Improving Helios with Everlasting Privacy Towards the Public. In *Electronic Voting Technology Workshop, 2012*. USENIX Association.
[27] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. 1992. A Practical Secret Voting Scheme for Large Scale Elections. In *AUSCRYPT 1992*. 244–251.
[28] Taher El Gamal. 1983. A Subexponential-Time Algorithm for Computing Discrete Logarithms over GF(p^2). In *CRYPTO 1983*. Plenum Press, 275–292.
[29] Huangyi Ge, Sze Yiu Chau, Victor E. Gonsalves, Huian Li, Tianhao Wang, Xukai Zou, and Ninghui Li. 2019. Koinonia: verifiable e-voting with long-term privacy. In *ACSAC 2019*. 270–285.
[30] Kristian Gjøsteen, Thomas Haines, and Morten Rotvold Solberg. 2020. Efficient Mixing of Arbitrary Ballots with Everlasting Privacy: How to Verifiably Mix the PPATC Scheme. In *NordSec 2020*. 92–107.
[31] Oded Goldreich. 2001. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press.
[32] Oded Goldreich. 2004. *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press.
[33] Panagiotis Grontas, Aris Pagourtzis, Alexandros Zacharakis, and Bingsheng Zhang. 2018. Towards Everlasting Privacy and Efficient Coercion Resistance in Remote Electronic Voting. In *FC 2018 International Workshops*. 210–231.
[34] Thomas Haines. 2019. Cronus: Everlasting Privacy with Audit and Cast. In *NordSec 2019*. 53–68.
[35] Thomas Haines and Xavier Boyen. 2016. VOTOR: Conceptually simple remote voting against tiny tyrants. In *ACM ACSWM 2016*. 32.
[36] Eduard Hauck, Eike Kiltz, Julian Loss, and Ngoc Khanh Nguyen. 2020. Lattice-Based Blind Signatures, Revisited. In *CRYPTO 2020*. 500–529.
[37] Lucca Hirschi, Lara Schmid, and David A. Basin. 2021. Fixing the Achilles Heel of E-Voting: The Bulletin Board. In *IEEE CSF 2021*. 1–17.
[38] Benjamin Hosp and Poorvi L. Vora. 2008. An information-theoretic model of voting systems. *Math. Comput. Model.* (2008), 1628–1645.
[39] Nicolas Huber, Toomas Krips, Ralf Küsters, Julian Liedtke, Johannes Müller, Daniel Rausch, Pascal Reisert, and Andreas Vogt. 2022. Kryvos: Publicly Tally-Hiding Verifiable E-Voting. In *ACM CCS 2022*.
[40] Vincenzo Iovino, Alfredo Rial, Peter B. Rønne, and Peter Y. A. Ryan. 2017. Using Selene to Verify Your Vote in JCJ. In *FC 2017 International Workshops*. 385–403.
[41] Ari Juels, Dario Catalano, and Markus Jakobsson. 2005. Coercion-resistant electronic elections. In *Workshop on Privacy in the Electronic Society - WPES 2005*. ACM, 61–70.
[42] Guillaume Kaim, Sébastien Canard, Adeline Roux-Langlois, and Jacques Traoré. 2021. Post-quantum Online Voting Scheme. In *FC 2021 International Workshops, Revised Selected Papers*. 290–305.
[43] Jonathan Katz and Yehuda Lindell. 2014. *Introduction to Modern Cryptography, Second Edition*. CRC Press.
[44] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. 2015. End-to-End Verifiable Elections in the Standard Model. In *EUROCRYPT 2015*. 468–498.
[45] Christian Killer, Markus Knecht, Claude Müller, Bruno Rodrigues, Eder J. Scheid, Muriel Figueredo Franco, and Burkhard Stiller. 2021. Æternum: A Decentralized Voting System with Unconditional Privacy. In *IEEE ICBC 2021*.
[46] Ralf Küsters, Julian Liedtke, Johannes Müller, Daniel Rausch, and Andreas Vogt. 2020. Ordinos: A Verifiable Tally-Hiding E-Voting System. In *IEEE EuroS&P 2020*. 216–235.
[47] Ralf Küsters, Johannes Müller, Enrico Scapin, and Tomasz Truderung. 2016. sElect: A Lightweight Verifiable Remote Voting System. In *IEEE CSF 2016*. 341–354.
[48] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2010. A Game-Based Definition of Coercion-Resistance and Its Applications. In *IEEE CSF 2010*. 122–136.
[49] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2010. Accountability: Definition and Relationship to Verifiability. In *ACM CCS 2010*. 526–535.
[50] Steven P. Lalley and E. Glen Weyl. 2014. Nash Equilibria for a Quadratic Voting Game. *CoRR* abs/1409.0264 (2014).
[51] Philipp Locher and Rolf Haenni. 2015. Verifiable Internet Elections with Everlasting Privacy and Minimal Trust. In *VoteID 2015*. 74–91.
[52] Philipp Locher and Rolf Haenni. 2016. Receipt-free remote electronic elections with everlasting privacy. *Ann. des Télécommunications* (2016), 323–336.
[53] Philipp Locher, Rolf Haenni, and Reto E. Koenig. 2016. Coercion-Resistant Internet Voting with Everlasting Privacy. In *FC 2016*. 161–175.
[54] David Mestel, Johannes Müller, and Pascal Reisert. 2022. How Efficient Are Replay Attacks Against Vote Privacy? A Formal Quantitative Analysis. In *IEEE CSF 2022*.
[55] Tal Moran and Moni Naor. 2006. Receipt-Free Universally-Verifiable Voting with Everlasting Privacy. In *CRYPTO 2006*. 373–392.
[56] Tal Moran and Moni Naor. 2007. Split-ballot voting: everlasting privacy with distributed trust. In *ACM CCS 2007*. 246–255.

[57] Kevin Morio and Robert Künnemann. 2021. Verifying Accountability for Unbounded Sets of Participants. In *34th IEEE Computer Security Foundations Symposium, CSF 2021, Dubrovnik, Croatia, June 21-25, 2021*. 1–16.
[58] Johannes Müller. 2022. Breaking and Fixing Vote Privacy of the Estonian E-Voting Protocol IVXV. In *Workshop on Advances in Secure Electronic Voting 2022*.
[59] Jörn Müller-Quade and Dominique Unruh. 2010. Long-Term Security and Universal Composability. *J. Cryptol.* (2010), 594–671.
[60] Torben P. Pedersen. 1991. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *CRYPTO 1991*. 129–140.
[61] Olivier Pereira and Peter B. Rønne. 2019. End-to-End Verifiable Quadratic Voting with Everlasting Privacy. In *FC 2019 International Workshops*. 314–329.
[62] Iñigo Querejeta-Azurmendi, David Arroyo Guardeño, Jorge L Hernández-Ardieta, and Luis Hernández Encinas. 2020. NetVote: A Strict-Coercion Resistance Re-Voting Based Internet Voting Scheme with Linear Filtering. *Mathematics* (2020), 1618.
[63] Ronald L. Rivest, Adi Shamir, and Yael Tauman. 2001. How to Leak a Secret. In *ASIACRYPT 2001*. 552–565.
[64] Peter Y. A. Ryan, Peter B. Rønne, and Vincenzo Iovino. 2016. Selene: Voting with Transparent Verifiability and Coercion-Mitigation. In *FC 2016 International Workshops*. Springer, 176–192.
[65] Peter Y. A. Ryan and Steve A. Schneider. 2006. Prêt à Voter with Re-encryption Mixes. In *ESORICS 2006*. 313–326.
[66] Peter Schwabe, Douglas Stebila, and Thom Wiggers. 2020. Post-Quantum TLS Without Handshake Signatures. In *ACM CCS 2020*. 1461–1480.
[67] Warren D. Smith. 2007. Three Voting Protocols: ThreeBallot, VAV, and Twin. In *USENIX/ACCURATE Electronic Voting Technology Workshop, 2007*.
[68] Björn Terelius and Douglas Wikström. 2010. Proofs of Restricted Shuffles. In *Progress in Cryptology - AFRICACRYPT 2010*. Springer, 100–113.
[69] Jeroen van de Graaf. 2009. Voting with unconditional privacy: CFSY for booth voting. *IACR Cryptol. ePrint Arch.* (2009).
[70] Jeroen van de Graaf. 2010. Anonymous One-Time Broadcast Using Non-interactive Dining Cryptographer Nets with Applications to Voting. In *Towards Trustworthy Elections, New Directions in Electronic Voting*. 231–241.
[71] Stefan G. Weber, Roberto Araújo, and Johannes Buchmann. 2007. On Coercion-Resistant Electronic Elections with Linear Work. In *IEEE ARES 2007*. 908–916.
[72] Nan Yang and Jeremy Clark. 2017. Practical Governmental Voting with Unconditional Integrity and Privacy. In *FC 2017 International Workshops*. 434–449.

A CRYPTOGRAPHIC PRIMITIVES

We recall the most common cryptographic primitives employed for secure e-voting and introduce our notation for these primitives. We refer to established text books (e.g., [31, 32, 43]) for formal definitions.

Digital signature. A *digital signature (DS) scheme* enables a party to sign messages so that everyone can convince herself that the signatures were created by that party but no-one else. We denote DS schemes by $\mathcal{S} = (\mathsf{KG}_s, \mathsf{S}, \mathsf{V}_s)$, where KG_s outputs a verification/signing key pair (vk, ssk), $\mathsf{S}(ssk, m)$ outputs a signature s, and $\mathsf{V}_s(vk, m, s)$ outputs a bit b. (Throughout this paper, we typically keep the security parameter 1^ℓ in the input to algorithms/processes implicit.)

Public-key encryption. A *public-key encryption (PKE) scheme* enables everyone to encrypt messages so that the content of the resulting ciphertexts can only be read by a designated receiver but no-one else. We denote PKE schemes by $\mathcal{E} = (\mathsf{KG}_e, \mathsf{E}, \mathsf{D})$, where KG_e outputs a public/private key pair (pk, sk), $\mathsf{E}(pk, m)$ outputs a ciphertext e, and $\mathsf{D}(sk, e)$ outputs a message m'. The most important security notions are *Chosen-Plaintext-Attack (CPA)* and *Chosen-Ciphertext-Attack (CCA)* security.

Some e-voting protocols employ *homomorphic* PKE schemes. In such schemes, ciphertexts $e_i = \mathsf{E}(pk, m_i)$ can be combined (without knowledge of the secret key sk or the messages m_i) to obtain a ciphertext e for which $\mathsf{D}(sk, e) = \sum_i m_i$; such schemes may be additively or multiplicatively homomorphic. A widely used homomorphic PKE scheme is ElGamal PKE [28], which is CPA-secure under the decisional Diffie-Hellman assumption.

Commitment. A *commitment scheme (CS)* enables everyone to commit to messages so that (1) anyone, who does not know how the resulting commitments were created, is unable to derive the original messages, and (2) everyone can be convinced by the committing party that the commitments in fact committed to the original messages. The first property is called *hiding*, and the second one *binding*. We denote CSs by $C = (\mathsf{KG}_c, \mathsf{C}, \mathsf{O})$, where KG_c outputs parameters prm, $\mathsf{C}(prm, m, r)$ outputs a commitment c, and $\mathsf{O}(prm, m, c, r)$ outputs a bit b.

Some e-voting protocols employ *homomorphic* CSs. In such schemes, commitments $c_i = \mathsf{C}(m_i, r_i)$ can be combined (without knowledge of the opening values (m_i, r_i)) to obtain a commitment $c = \mathsf{C}(\sum_i m_i, \sum_i r_i)$. A widely used homomorphic CS is Pedersen's CS [60], which is unconditionally hiding and computationally binding under the discrete logarithm assumption.

Zero-knowledge proof. A *non-interactive zero-knowledge proof (NIZKP) system* enables a party to prove non-interactively that a certain statement holds true without revealing any information beyond the correctness of that statement. The property that ensures that a dishonest prover cannot create a valid proof for a false statement is called *soundness*; the property that no more information can be extracted from the proof than the correctness of the statement is called *zero-knowledge*. We denote NIZKP systems by $\Pi = (\mathsf{P}, \mathsf{V})$, where the prover P takes as input the statement/witness pair (s, w) of the respective relation and outputs a proof π, and the verifier V takes as input (s, π) and outputs a bit b.

Many e-voting protocols employ a NIZKP *of knowledge*, which requires that a prover also needs to *know* a witness of the respective statement.

B COMPARISON OF SECRET-SHARING SOLUTIONS

Cramer *et al.*	**Ge *et al.***	**Alternative**
V_i: $(v_i^j)_j \leftarrow \mathsf{Share}(v_i)$, $c_i^j \leftarrow \mathsf{C}(prm, v_i^j, r_i^j)$, $\pi_i \leftarrow \mathsf{P}((prm, c_i), (v_i, r_i))$		
Send (v_i^j, r_i^j) to T_j	Send (c_i^j, v_i^j, r_i^j) to T_j T_j: $s_i^j \leftarrow \mathsf{S}(ssk_j, c_i^j)$	$e_i^j \leftarrow \mathsf{E}(pk_j, (v_i^j, r_i^j))$
$b_i \leftarrow (i, (c_i^j)_j, \pi_i)$	V_i: $\lightning$ if $\mathsf{V}_s(vk_j, c_i^j, s_i^j) = 0$ $b_i \leftarrow (i, (c_i^j)_j, \pi_i, (s_i^j)_j)$	$b_i \leftarrow (i, (c_i^j)_j, \pi_i, (e_i^j)_j)$
Send b_i to PBB		Send b_i to SBB SBB: $h_i^j \leftarrow \mathsf{H}(e_i^j)$ $b_i' \leftarrow (i, (c_i^j)_j, \pi_i, (h_i^j)_j)$ Send b_i' to PBB $\mathsf{V}_i, \mathsf{T}_j$: $\lightning$ if $h_i^k \neq \mathsf{H}(e_i^k)$ T_j: $(v_i^j, r_i^j) \leftarrow \mathsf{D}(sk_j, e_i^j)$
T_j: $\lightning$ if $\mathsf{O}(prm, c_i^j, v_i^j, r_i^j) = 0$ or $\mathsf{V}((prm, c_i), \pi_i) = 0$		
	$\lightning$ if $\mathsf{V}_s(vk_k, c_i^k, s_i^k) = 0$	

Figure 1: The voting phase in solutions based on secret-sharing (see Section 6.2). Share **denotes the share algorithm of a secret-sharing scheme,** H **a (cryptographic) hash function,** $\lightning$ **a complaint on** PBB**.**