TRIP: Coercion-resistant Registration for E-Voting with Verifiability and Usability in Votegral

Preliminary draft - final version to appear in SOSP 2025

Louis-Henri Merino
EPFL
Switzerland

Simone Colombo
King's College London
United Kingdom

Rene Reyes
Boston University
USA

Alaleh Azhir
Harvard University
USA

Shailesh Mishra
EPFL
Switzerland

Pasindu Tennage
EPFL
Switzerland

Mohammad Amin Raeisi
EPFL
Switzerland

Haoqian Zhang
EPFL
Switzerland

Jeff Allen
EPFL
Switzerland

Bernhard Tellenbach
Armasuisse S+T
USA

Vero Estrada-Galiñanes
EPFL
Switzerland

Bryan Ford
EPFL
Switzerland

Abstract

Online voting is convenient and flexible, but amplifies the risks of voter coercion and vote buying. One promising mitigation strategy enables voters to give a coercer fake voting credentials, which silently cast votes that do not count. Current proposals along these lines make problematic assumptions about credential issuance, however, such as strong trust in a registrar and/or in voter-controlled hardware, or expecting voters to interact with multiple registrars. Votegral is the first coercion-resistant voting architecture that leverages the physical security of in-person registration to address these credential-issuance challenges, amortizing the convenience costs of in-person registration by reusing credentials across successive elections. Votegral's registration component, TRIP, gives voters a kiosk in a privacy booth with which to print real and fake credentials on paper, eliminating dependence on trusted hardware in credential issuance. The voter learns and can verify in the privacy booth which credential is real, but real and fake credentials thereafter appear indistinguishable to others. Only voters actually under coercion, a hopefully-rare case, need to trust the kiosk. To achieve verifiability, each paper credential encodes an interactive zero-knowledge proof, which is sound in real credentials but unsound in fake credentials. Voters observe the difference in the order of printing steps, but need not understand the technical details. Experimental results with our prototype suggest that Votegral is practical and sufficiently scalable for real-world elections. User-visible latency of credential issuance in TRIP is at most 19.7 seconds even on resource-constrained kiosk hardware, making it suitable for registration at remote locations or on battery power. A companion usability study indicates that TRIP's usability is competitive with other E-voting systems including some lacking coercion resistance, and formal proofs support TRIP's combination of coercion-resistance and verifiability.

ACM Reference Format:
Louis-Henri Merino, Simone Colombo, Rene Reyes, Alaleh Azhir, Shailesh Mishra, Pasindu Tennage, Mohammad Amin Raeisi, Haoqian Zhang, Jeff Allen, Bernhard Tellenbach, Vero Estrada-Galiñanes, and Bryan Ford. 2025. TRIP: Coercion-resistant Registration for E-Voting with Verifiability and Usability in Votegral: Preliminary draft - final version to appear in SOSP 2025. In *Proceedings of the 31st ACM Symposium on Operating Systems Principles (SOSP '25), October 13–16, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 48 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Democracy is the operating system of free self-governing civilization: it holds governments accountable to their citizens, and enables correction and renewal by peaceful transfer of power: periodically "rebooting" the government, so to speak. Tech-savvy idealists often dream that large-scale distributed computing systems could improve or even positively transform democracy, enabling more convenient large-scale direct participation via mechanisms like liquid democracy [32, 62] or online citizens' assemblies [91]. However, today's computing systems are not even *safe to use* in basic democratic processes – such as popular voting – according to the overwhelming consensus of today's experts [6, 19, 133].

A central challenge for national voting systems is resolving the conflicting goals of transparency and ballot secrecy [24, 27, 60, 78, 110]. Transparency requires convincing voters that their ballots were cast and counted properly, while ballot secrecy ensures that voters can express their true preferences, free from improper influence such as coercion or vote buying. In the best practice of in-person voting, voters get ballot secrecy by marking paper ballots alone in a privacy booth at an official polling site. Voters obtain "end-to-end" transparency by first depositing their marked ballots in a ballot box, then relying on election observers to monitor the subsequent handling and tallying of all the ballots.

While in-person voting is widely accepted by experts [6], it has posed significant inconveniences to voters, including long wait times [56, 139], temporary polling place closures [14], and even voter intimidation near polling locations [139]. In-person voting is additionally difficult for those traveling and living abroad, such as expatriates or deployed military [127] and during crises like the recent pandemic [137].

Remote voting—at any location of the voter's choosing—promises to improve convenience and increase voter turnout [117, 127]. Because the voting location is uncontrolled, however, remote voting normally compromises ballot secrecy and hence resistance to coercion or vote-buying [11, 36, 59, 66, 68, 121, 139]. An abusive spouse might insist on their partner voting their way under supervision, a party activist might offer to purchase and "help" fill mail-in ballots, as seen in the United States [9, 70], or a foreign power might attempt the same at scale, as seen recently in Moldova [11]. Postal voting also compromises transparency by subjecting ballots to unpredictable delays and potential loss [7, 76, 83].

Remote electronic voting or *e-voting*, using a voter's preferred device, promises further convenience benefits and avoids postal issues [53, 109, 127]. E-voting can even offer voters greater transparency via "end-to-end" cryptographic proofs, verifiable on their device, that their vote was cast and tallied correctly [16, 24, 42]. However, such proofs, or *receipts*, can also enable voters to prove how they voted to a coercer or vote buyer [27, 78, 110]. "Dark DAOs" might even fund vote-buying anonymously, at scale and across borders, with no realistic prospect for accountability or deterrence [21].

One seminal proposal to counter coercion in e-voting is to provide voters with both real and *fake credentials* [78]. Only the voter knows which credential is real, enabling them to give or sell fake credentials to a coercer or vote buyer while secretly casting their true vote using the real credential. A key limitation of this appealing idea, however, is that it not so much *solves* but rather *shifts* the most security-critical, delicate, and challenging stage from voting time to credential-issuance or *registration* time. Efforts to make fake credentials practical often make strong and arguably-unrealistic assumptions that all voters have special trusted hardware, such as expensive smart cards, at registration [41, 55, 58, 104]. A common implicit assumption is that a coercer cannot simply confiscate the voter's trusted hardware and allow its use only under their supervision. Related approaches either compromise transparency by assuming a fully trusted registrar [78], or achieve transparency at the cost of requiring voters to interact with multiple registration authorities [41] – a usability challenge that has yet to be rigorously studied.

We present Votegral, the first coercion-resistant online voting system with end-to-end verifiability and evidence of potential usability. This paper focuses on TRIP, the registration component of Votegral. TRIP is a novel registration system enabling voters to obtain verifiable real and *indistinguishable* fake voting credentials on paper. TRIP addresses three key challenges: 1) issuing verifiably real voting credentials without requiring voters to have a trusted device during registration, 2) issuing fake credentials that are distinguishable only to the voter, and 3) materializing these real and fake credentials usably with only inexpensive paper materials.

TRIP leverages an in-person process to give voters a coercion-free environment in which to create voting credentials. Unlike in-person voting, registration may be scheduled at the voter's convenience, and voting credentials can cast votes in multiple successive elections. To address the first challenge of issuing verifiable real credentials, voters interact with a kiosk in a privacy booth. The kiosk proves to the voter that this credential is in fact real using an *interactive* zero-knowledge proof (IZKP), although the voter need not understand the details. To address the second challenge of creating fake credentials distinguishable only to the voter, the voter and the kiosk follow a visibly distinct process in which the kiosk forges a *false* IZKP that is subsequently indistinguishable from the real one. The voter thus knows which credential is real but is unable to prove that fact to anyone, and can safely give or sell fake credentials to a coercer.

To address the third challenge of materializing real and fake credentials, kiosks print all credentials on paper. Paper-based credentials are inexpensive, making it more easy and cost-effective for most voters to create a few fake credentials including for reasons other than coercion risks. Voters can activate credentials on any device they choose, including a trusted friend's device if their own is under a coercer's control. Printed envelopes supplied to voters in the booth form a part of each credential, simplifying the voter's task of choosing a random challenge for each IZKP, and serving to conceal sensitive secret keys during credential transport.

While this paper focuses on e-voting, we view our work as a stepping-stone toward secure large-scale *democratic computing systems* usable by self-governing groups, such as user communities, organizations, and nations. For example, many decentralized projects wish to incorporate democratic self-governance into their designs with decentralized autonomous organizations but lack the means to ensure that voting participants are *real people* acting in their own interest [129]. Integrating TRIP's coercion-resistance mechanism into in-person *pseudonym parties* as a "proof-of-personhood" protocol [34] could help address this challenge and enable truly democratic DAOs. Appendix A further discusses the broader potential systems applications of this work.

We implemented a prototype of Votegral and TRIP consisting of 2,557 lines of Go [98]. Our Votegral and TRIP prototypes focus on the cryptographic path, which represents the dominant computation and latency cost in e-voting systems. We evaluated Votegral against three state-of-the-art e-voting systems: (1) Civitas, an end-to-end verifiable and coercion-resistant system based on fake credentials [41], (2) the Swiss Post's verifiable but non-coercion-resistant system [8], and (3) VoteAgain, a coercion-resistant but not end-to-end verifiable system based on deniable re-voting [96]. We find that Votegral's end-to-end latency is comparable to that of Swiss Post and VoteAgain, and significantly improves upon Civitas. For TRIP, we also implement the use of peripherals to determine TRIP's voter-observable latency across several setups: (1) a Point of Sale Kiosk, (2) a Raspberry Pi 4, (3) a Macbook Pro, and (4) a Mini PC. We find that TRIP's voter-observable latency is slowest on the Kiosk at 19.7 seconds and fastest on the Macbook Pro at 15.8 seconds, both suitable inside a booth where voters typically spend a few minutes.

To refine and validate our design, we incorporate feedback from two preliminary user studies involving 77 participants. We also summarize key insights from our main user study involving 150 participants, which we conducted on this design [97]. We find that 83% of participants successfully created and used their real credential to cast a mock vote. Additionally, 47% of participants exposed to a malicious kiosk, with security education, could identify and report it.

We summarize important limitations of Votegral in §4.7.

This paper makes the following primary contributions:

- TRIP, the first coercion-resistant, user-studied voter registration scheme to offer a concrete realization of real and fake credentials while maintaining verifiability.
- The first use of paper transcripts of interactive zero-knowledge proof to achieve verifiability during registration, avoiding reliance on trusted hardware.
- Security proofs demonstrating that TRIP satisfies verifiability and coercion-resistance.
- An implementation and evaluation of TRIP against state-of-the-art baselines on multiple hardware platforms.

2 Background and motivation

Central requirements for a national voting system include transparency, ballot secrecy, and usability [24, 27, 60, 78, 110]. The importance of transparency in elections is evident from the contrastingly non-transparent *sham elections* routinely held by dictatorships to support predetermined outcomes [138, 141]. Ballot secrecy was first recognized gradually across several countries, but quickly became standard internationally after Australia's introduction of the modern secret ballot in 1856 [44]. Moreover, election systems must be comprehensible and reliably usable by all voters – including those with special needs or disabilities – to ensure broad, equitable participation [119]. Despite efforts towards digitalizing voting, in-person voting with paper ballots remains the accepted best practice, as further discussed in Appendix B.

2.1 Remote voting with end-to-end verifiability

E-voting systems aim to enhance transparency by providing cryptographic end-to-end verifiability of the voting and tallying process [8, 16, 43, 90]. In a simplistic but illustrative sketch of such a process, the voter's personal device encrypts the voter's choices and posts the encrypted ballot to a *public bulletin board* or PBB that anyone can read, such as a tamper-evident log [45] or blockchain [140]. Each encrypted ballot is posted along with voter identity information sufficient to ensure eligibility and prevent multiple votes. After the voting deadline, each of several *talliers* reads the ballot ciphertexts from the PBB, shuffles them to anonymize the ballots (scrubbing the linkage to the voter's identity), then strips off one layer of ballot encryption, and posts the shuffled ballots back onto the PBB along with a zero-knowledge proof that the shuffle was performed correctly [71, 102]. After these shuffles and decryptions, anyone may tally the cleartext ballots left on the PBB and verify the series of proofs that they correspond to the originally-cast ballots, providing *universal verifiability* of the process. Each voter may additionally use their private encryption material to check that their choices were correctly encrypted and recorded on the PBB, and therefore ultimately tallied in this process.

While this e-voting sketch offers individual and universal verifiability, it unfortunately compromises ballot secrecy by effectively giving voters a *receipt* [27, 78], allowing them to prove how (and whether) they voted. A coercer need merely check the PBB to see whether the voter cast a ballot, and if so, demand the voter's receipt to confirm compliance.

2.2 Coercion resistance in e-voting systems

Most proposals to address coercion in e-voting rely on either re-voting or fake credentials. Re-voting lets voters override coerced votes by casting their intended vote later in secret [15, 96, 112, 116]. Estonia is currently the only nation to deploy an e-voting system with coercion-resistance using the re-voting approach [53], but it comes at the cost of end-to-end verifiability [133]. Moreover, re-voting can be circumvented by coercing a last-minute vote, or by seizing key voting materials – such as the voter's ID card used in Estonia's system [2, 10] – until after the election.

The *fake credentials* approach [78] instead relies on enabling voters to obtain both real and fake credentials. A voter may use a fake credential under a coercer's supervision, or give or sell fake credentials to a coercer or vote-buyer. A voter may cast their intended vote in secret at any time with their real credential, avoiding re-voting's weakness of requiring the real vote to follow all coerced votes. A limitation of

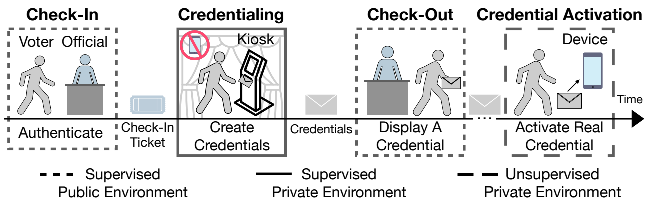


Figure 1. TRIP Voter Workflow
The voter (1) authenticates themselves to an official at a check-in desk to obtain a check-in ticket (2) enters a supervised private environment to create a real and any potential fake credentials, (3) presents one of their credentials to an official at the check-out desk, and (4) activates their real and any fake credentials on a device they trust, which could belong to them or a trusted third party.

this approach, however, is that it time-shifts key unsolved problems earlier from voting to registration time.

Making fake credentials practical requires physically embodying or *materializing* credentials and their issuance, in a way that ordinary people can understand and use [88, 104, 105]. One key challenge is that the voter must know which credential is real, and must personally be able to *verify* it as real for transparency, but then must be unable to *prove* this fact to anyone else. A coercer must also not know how many fake credentials the voter has or can create; otherwise a coercer can simply demand *all* of the voter's credentials.

Usability-focused approaches to materializing fake credentials have generally followed Estonia's lead by assuming trusted hardware, such as smart cards that store both real and fake credentials under different PINs [41, 58, 87, 104, 105]. Even under the as-yet-unproven assumption that sufficiently-powerful and secure smart cards can be developed, they will be expensive, making even one per voter difficult to justify economically – let alone *several* per voter, as would be required to achieve real coercion resistance. If governments issue only one smart card per voter, then, as with Estonia's e-ID cards, a coercer can simply "offer" to "keep safe" (i.e., confiscate) the voter's smart card and allow its use only under supervision, negating its effective coercion resistance.

Registration-time transparency is another key challenge. Why can't a compromised election authority just issue fake smart cards, for example, which silently issue *only* fake credentials, while the authority retains the secret keys required to cast "real" votes on behalf of all voters?

3 Votegral Design Overview

This section provides a high-level overview of Votegral's architecture to provide context needed to understand TRIP. We defer a detailed technical description of TRIP, the focus and central contribution of this paper, to §4.

3.1 High-Level election process

Any voting system generally includes *setup*, *voting*, and *tallying* phases. In the setup phase, the election authority establishes the *roster* or list of eligible voters, and the set of choices that each voter is entitled to make, then prepares the system to accept ballots. In the voting phase, voters cast ballots via one or more voting *channels* or ballot-submission methods. For clarity we will assume here that e-voting via Votegral is the only channel, but Appendix C outlines some considerations applicable if Votegral were to be integrated with other channels such as in-person and/or postal voting. Once the voting deadline passes, the system accepts no more ballots and the tallying phase begins, in which all properly-submitted ballots are counted and the results published. In systems like Votegral supporting fake credentials, it is crucial that the tallying phase count only votes cast using real credentials.

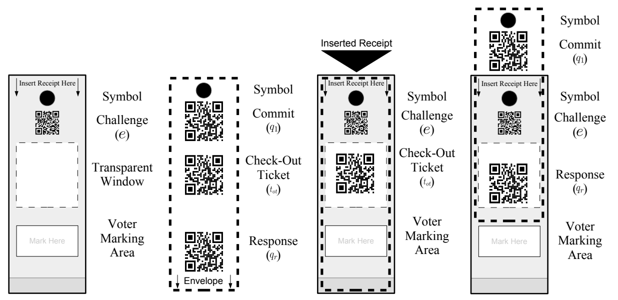
Registration. Like any coercion-resistant design with fake credentials, Votegral assumes that voters *register* for e-voting before they can cast ballots. Votegral more uniquely requires that this registration for e-voting be done in person. Votegral's *registration for e-voting* may, but need not necessarily, coincide with *registration to vote*, as further detailed in Appendix C. In US-style elections where voters must register to vote anyway, voters might in principle register for e-voting via Votegral at the same time. In Europe-style elections where registration *to vote* is normally automatic, registration *for e-voting* might be a separate step required only of those wishing to opt-in to e-voting. Regardless, we expect that registration for e-voting (or to vote) should normally be required at most once every several elections, thus amortizing the cost of in-person registration across successive uses of the same credentials.

In Votegral's design, we consider in-person registration to not just be a step technically needed to create real and fake credentials, but also an educational opportunity for voters to learn and ask questions about the e-voting system, and an opportunity for voters to report and discuss actual attempts at coercion or other voting irregularities in a protected environment. We will focus here on the actual process of credential creation, however – first from the voter's perspective, ignoring technical details, in the next subsection.

Renewal. While we expect TRIP credentials to be reusable across successive elections, they will normally expire at some point, and need to be renewed by again registering in-person. While credential lifetime is a policy choice outside this paper's scope, Appendix D discusses some credential lifetime considerations, including the tension between voter convenience and recovery from "surprise coercion" situations.

3.2 Voter-facing design of e-voting registration

Registering to use Votegral involves the following main steps, which Fig. 1 illustrates at a high level.



(a) Envelope **(b)** Receipt **(c)** Transport **(d)** Activate

Figure 2. Receipt and Envelope Design.
A paper credential consists of an envelope (a) and receipt (b). Voters mark envelopes to discern their real credential from fake ones. They transport each credential by placing the receipt into the envelope (c) and activate it by lifting the receipt one-third of its length (d).

Instructional Video. In the registrar's office, voters first watch an instructional video, covering the credential creation processes and the purpose of fake credentials.
Check-In. A registration official verifies each voter's eligibility, then gives the voter a *check-in ticket*, which permits the voter to enter a privacy booth and use the kiosk inside. Depending on applicable anti-recording policy, voters might be asked to turn off or check in personal devices before entering the privacy booth, as further discussed in Appendix E.
Privacy Booth. Inside the booth is a touchscreen kiosk that can scan and print machine-readable codes, along with a pen and a large supply of envelopes. Each envelope has a see-through area and printed markings as depicted in Fig. 2a.
Real Credential Creation. Voters follow a 4-step process to create their real credential. The voter first scans the barcode on their check-in ticket (Step 1). The kiosk prints a symbol and a QR code on receipt paper (Step 2). The voter selects an envelope with a matching symbol and scans its QR code (Step 3). The kiosk then prints two additional QR codes (Step 4), completing the receipt shown in Fig. 2b. The voter inserts the receipt inside the scanned envelope for transport (Fig. 2c), forming their real paper credential. The voter marks the credential, for instance by labeling it as '$\mathcal{R}$', to distinguish it from any fake credentials. If a coercer learns or guesses that the voter does this, then at the next registration the voter can mark a fake credential '$\mathcal{R}$' and mark their real credential '$\mathcal{RR}$' or in any memorable way.
Fake Credential Creation. A voter wishing to create a fake credential does so in 2 steps: the voter picks and scans an envelope (Step 1, Fig. 2a), then the kiosk prints the entire receipt (Step 2, Fig. 2b). The voter inserts the receipt into the newly-chosen envelope and marks it differently from the real one. Voters can create as many fake credentials as desired. Officials may inquire if a voter spends excessive time in a booth, thus imposing an informal, non-deterministic limit. Once finished, the voter leaves the booth and checks out.

Check-Out. The voter presents any one of their credentials to the official, who scans the receipt's second QR code through the envelope's window. This completes the voter's visit to the registrar. To ensure prompt detection and rectification of any successful impersonation of voters, the registrar subsequently notifies voters digitally and/or by mail of the completed registration session, as discussed in Appendix F.
Activation. The voter activates their real credential on any device they trust, whether that is their own or a friend's device. The voter lifts the receipt one-third out of the envelope to the *activate* state (Fig. 2d). They scan the three visible QR codes: the receipt's top and bottom QR codes, and the envelope's QR code. The voter can now discard this credential, perhaps shredding it if the voter faces a coercion threat.
Voting. The voter can now use this device to cast their ballot, verifying it with existing cast-as-intended methods [25, 42] to ensure its integrity. Only the vote cast with the real credential will count, however. A voter who loses their device can re-register for new credentials. Other options like social key recovery are also possible as discussed in Appendix G.
Results. After each election, the voter's device downloads, verifies, and shows the election results to the voter.

3.3 Desired System Properties

We now informally present Votegral's system properties.

- **Universal Verifiability** [8, 78]: anyone can independently verify that the election results accurately reflect the will of the electorate as represented by the ballots cast.
- **Individual Verifiability** [8, 29]: each voter can independently verify that their issued real credential casts ballots that are accurately recorded and included in the final tally, while verifying that this ballot reflects their intended vote.
- **Coercion Resistance** [78]: voters can cast their intended vote even when faced with coercion because the coercer cannot determine whether the voter has complied with the coercer's demands, even if the voter wishes to comply.
- **Privacy** [78]: uncompromised voters' ballots stay secret.
- **Usability** [51, 88, 107]: voters can understand and use the voting system, including for registering and casting ballots, and can understand the uses of real and fake credentials.

End-to-end verifiability requires both individual and universal verifiability, with voters verifying the integrity of their own real vote, and the public at large verifying the results.

4 Registration and Voting Protocol Design

This section informally describes the Votegral protocol, with a particular focus on TRIP, the system's registration stage.

The core design of Votegral addresses four key challenges: (1) threat modeling and designing a registration process in which we trust neither voters *nor* registrars in general; (2) providing an end-to-end verifiable registration, voting, and tallying process despite only voters themselves knowing how many credentials they have or which of them is real;

(3) enabling the registration kiosk to prove interactively to a voter that a credential is real without the voter being able to prove that fact subsequently to a coercer; and (4) materializing this credentialing and interactive proof process in a usable fashion using only low-cost disposable materials such as paper. The rest of this section addresses these challenges, then §4.6 summarizes optional extensions to the base design.

4.1 System model and threat model

We summarize Votegral's system and threat models here at a concise intuitive level, leaving detailed formal system model and threat models used in our proofs to Appendix I.

Consistent with common e-voting system designs, we assume an *election authority*, or simply *authority*, responsible for setting up and coordinating the election process. The registration sites or *registrars* at which voters can sign up for e-voting answer to, or at least coordinate with, the election authority. The authority provides client-side software that voters need to activate credentials and cast ballots; for transparency we assume this software is open source. Finally, the authority is responsible for providing two logical backend components: a *ledger* or *public bulletin board* implementing a tamper-evident log [46], and a *tally service*.

We assume that these backend services are distributed across multiple servers, preferably operated independently, ensuring that backend services remain secure provided not all servers are compromised. Backend trust-splitting is standard in cryptography and e-voting practice and not novel or unique to Votegral: e.g., the Swiss e-voting system splits backend services across four "control components" maintained by independent teams [113]. For exposition clarity, we leave this trust-splitting implicit for now, as if the ledger and tally services were trusted third parties. In contrast with Civitas' expectation that users interact directly with multiple registrars [41], splitting backend services does not introduce usability issues because ordinary users need not interact directly with them, nor even be aware of them. Users interact with backend services only through the front-end client, which anyone with sufficient expertise may inspect but ordinary voters are not required or expected to understand.

One challenge to threat modeling Votegral is that we wish to trust neither voters *nor* registrars in general, but if we distrust *all voters* and *all registrars* at once then we have no foundation on which to build any arguably-secure system. We address this challenge by specializing the threat model to distinct threat vectors, which we model as separate adversaries. To evaluate Votegral's transparency we use an *integrity adversary* $\mathcal{I}$, which can cause any or all registrars to misbehave arbitrarily – attempting to steal voters' real credentials and issue only fake credentials for example – but which prefers not to be "caught" in such misbehavior. To evaluate Votegral's coercion resistance we use a *coercion adversary* C, which can coerce and hence effectively compromise voters, but which cannot compromise registrars. The

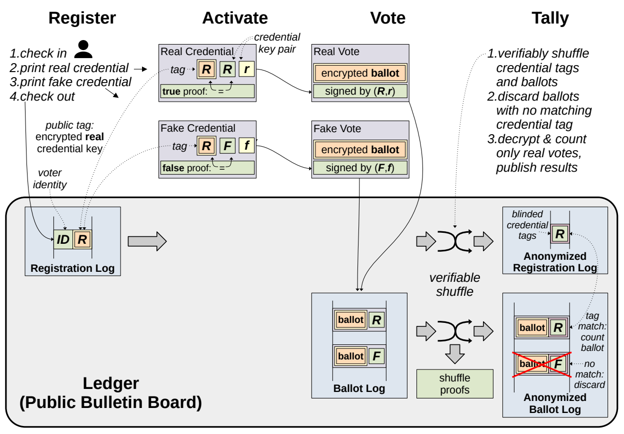


Figure 3. Votegral Protocol Design Summary

key desirable property this nuanced threat model achieves is that the majority of voters *not* under coercion need not trust the registrars in order to verify the end-to-end correctness of the election. Only the hopefully-few voters *actually under coercion* must trust the registrar. This trust appears unavoidable because in this case we must treat the voter himself as untrustworthy, by virtue of being coerced or incentivized to follow the coercer's demands.

Our formal analysis also considers a *privacy adversary*, whose aim is to identify and decrypt a voter's real ballot; unlike the coercion adversary, however, this adversary cannot directly compromise or exert coercive influence over voters.

4.2 High-level protocol flow summary

Figure 3 presents an illustration of Votegral's overall protocol operation, omitting various details for clarity and focusing on a single voter, Alice, who creates one real and one fake credential. Alice first registers using the voter-facing process above in §3.2, then at home activates her real and fake credentials on her voting device(s). At the next election, Alice uses each credential (on the same or different devices) to cast real and fake ballots. After the election, the tally service anonymizes all ballots, discards those cast via fake credentials, and counts the remaining ballots, posting the results.

Upon Alice's registration, her registrar posts a record to the *registration log* on the ledger, including metadata uniquely identifying Alice as an eligible voter. This identity metadata might include Alice's name in districts where voting rosters are considered public, or might just be a pseudonymous index into the authority's eligible voter database if the database is not public. Each time Alice registers or renews, the new record supersedes and invalidates all prior registration records for the same voter identity. Thus, there is always at most one active registration record per voter, and for transparency anyone can at least count the total number of active records and check it against relevant census data, regardless of how much metadata about each voter may be public.

Each credential in TRIP has a unique public/private key pair (K, k). In this example, Alice's real credential has key pair (R, r), and her fake credential has a different key pair (F, f). The main use of each credential's key pair is to authenticate – essentially as if by signing – encrypted ballots submitted using that credential. Each credential also has a *public credential tag*, which unlike the per-credential key pairs, is identical across all credentials created in the same registration session. In all of Alice's new credentials, this public tag is an encryption of the public key associated with Alice's *real* credential. This tag goes into Alice's registration record on check-out. It does not matter which credential Alice shows on check-out because all have this same tag.

After the election deadline, Votegral's tally service uses verifiable shuffles [71] in a mix cascade [50] to anonymize the set of public credential tags associated with currently-active records in the registration log. The tally service similarly shuffles the set of all encrypted ballots cast using any credential. In this shuffling process, the tally service cryptographically blinds the public key that each credential was submitted with, and in parallel, decrypts while identically blinding all the public credential tags derived from the registration log. Because the public tag on all of Alice's credentials was an encryption of the public key of Alice's *real* credential, this means that the blinded credential key associated with Alice's real vote now matches the decrypted and blinded public credential tag derived from the registration log. The tally service verifiably counts Alice's real vote as a result of this tag match. The credential public keys associated with fake votes, however, do not match any blinded public credential tag after the shuffle, and as a result are discarded.

The design sketched so far mostly achieves universal verifiability. Anyone may inspect the eligible voter roster on the registration log, and the encrypted ballots on the ballot log; anyone can check the shuffle proofs to verify that both the roster and ballots were shuffled via a correct but secret permutation; and, anyone can verify that only ballots authorized by appropriate registration log entries are counted, at most one ballot per voter. We have not yet achieved individual verifiability, however: how does Alice know or verify which of her credentials is real, or that any of them are?

4.3 Interactively proving real credentials real

To enable Alice to verify individually that the ballot she cast with her real credential will be counted, she needs to "know" that the public credential tag printed on all her credentials and recorded in the registration log is a correct encryption of her real credential's public key. This is ultimately the critical single bit of information Alice needs to complete the end-to-end transparency chain and verify that her real vote will count. For coercion resistance, however, Alice must be unable to prove that critical information bit to anyone else.

As a straw-man, the kiosk might give Alice an ordinary non-interactive zero-knowledge proof, of the kind commonly used in digital signatures, that her credentials' public tag is an encryption of her real credential. Alice could then be certain that her real credential is real – but then she might also take that proof (receipt) with her upon leaving the booth and hand it to a coercer or vote buyer, who would be equally convinced. Such a non-interactive proof would therefore support transparency but undermine coercion resistance.

A key observation is that we can use *interactive* zero-knowledge proofs or IZKPs to convince Alice that her real credential is real without making that information transferable. TRIP uses IZKPs taking the common Σ*-protocol* form, which consist of three steps: *commit*, *challenge*, and *response*. For such an IZKP to be sound and effectively prove anything, the protocol must be executed in precisely this order: *first* the prover chooses a cryptographic commitment; *then* the verifier chooses a challenge value previously-unknown to the prover; *finally* the prover computes the sole valid response corresponding to the combination of commit and challenge. If the Σ-protocol is executed in the wrong order – in particular, if the prover knows or can guess the verifier's challenge *before* choosing the commit – then the IZKP is unsound and the prover can trivially "prove" anything.

Leveraging this observation, TRIP includes in all credentials, real and fake, a *transcript* of the three phases of an IZKP, which *purports* to prove that the public credential tag contains an encryption of the credential's public key – i.e., that the credential is real. Anyone can verify the structural validity of these IZKP proof transcripts, and voters' devices do so automatically when activating credentials. Such a transcript alone omits one crucial bit of information, however: was the commit, or the challenge, chosen first? TRIP achieves coercion-resistant verifiability by revealing this one crucial bit of information to the voter only interactively, via the sequence of steps the voter takes in the privacy booth to print real and fake credentials. Once the voter observes this distinction in printing steps, the credentials themselves embody only IZKP transcripts, which lack the crucial bit of "voter's-eyes-only" information, and are useless to prove to anyone else which credential is real. Real credentials contain sound IZKPs, fake credentials contain unsound ones, and the two are cryptographically indistinguishable once printed.

We are not the first to use IZKPs for coercion resistance along these lines; similar ideas have been explored for in-person voting [99]. To our knowledge, however, TRIP is the first design to use IZKPs in *registratiion* for e-voting to achieve verifiability and coercion resistance using fake credentials.

4.4 Physically materializing usable credentials

Although IZKPs in principle resolve the conflict between individual verifiability and coercion resistance, for usability we cannot expect ordinary voters to understand e-voting systems or cryptography, or any of the technical details underlying credentials, or precisely *why* technically Alice should be convinced that her real credential is real. In particular, we need a physically *materialized* credentialing process that ordinary voters can understand and follow given only minimal training, and ideally that is cheap and efficient enough to deploy at scale. These are the considerations that motivate TRIP's specific design using paper credentials.

One particular issue is that to ensure that the kiosk *must* honestly produce sound IZKPs for real credentials, the voter – taking the verifier's role in the IZKP – must choose and give the kiosk a cryptographic challenge, only *after* the kiosk has printed the IZKP commit. Choosing and entering a high-entropy random challenge manually would be tedious and burdensome, and ordinary users are bad at choosing random numbers [106]. These considerations motivate TRIP's choice to use envelopes, each pre-printed with a unique random QR code, enabling voters to pick a challenge by selecting any envelope from a supply provided in the booth. A compromised registrar could duplicate envelopes to make challenges more predictable, but an activation-time check for duplicate challenges detailed in Appendix L.3.5 ensures high-probability detection if many envelopes are duplicated.

Above and beyond the instructional video, we would like the credentialing process itself to help train voters to learn and expect the correct process, especially for printing real credentials. To ensure in particular that voters do not hastily choose or present an envelope too early, before the kiosk has printed the IZKP commit, an honest kiosk first chooses one of a few symbols at random and prints it just above the commit QR code; the voter must then choose any envelope with the matching symbol. When interacting with an honest kiosk, therefore, the voter *must* wait to see the symbol (and commit) printed before picking an envelope, otherwise the honest kiosk gently rejects the voter's choice of envelope. If a voter accustomed to this process later encounters a compromised kiosk that asks them to present an envelope first while supposedly printing a "real" credential, we hope that the voter's normal-case training will make such an irregularity more noticeable.

To create fake credentials, in contrast, the kiosk asks the voter to choose and present an envelope first. The kiosk thus obtains the challenge before choosing its commit, thus enabling the kiosk to fake a "proof" of the credential's "realness." Again, we hope that even voters completely unaware of the reasons for this curious process will nevertheless be able to follow it and, at least with moderate probability, notice and report if a compromised kiosk ever attempts to use the fake-credential process to print a "real" credential.

TRIP's envelope and receipt design secures transport and activation. Fully inserting the receipt into the envelope places the credential in the *transport* state (Fig. 2c). In this state, the QR code necessary for check out appears through the envelope's window, but the envelope's opaque lower portion hides the area of the receipt containing the credential's secret key. Only after the voter transports the credential to whatever device the voter trusts to activate it on, the voter pulls the receipt out of the envelope just enough to reveal the two other QR codes needed for activation, including the credential's secret key, as shown in Fig. 2d.

4.5 Further design subtleties

This section briefly points out a few subtleties to TRIP's design not captured in the above design summary.

Impersonation defenses. To ensure prompt detection and remediation of any successful registration-time impersonation of a voter – whether by a look-alike or by a corrupt registration official – TRIP notifies voters of all registration events. Appendix F presents in more detail these threats, defenses, and their implications for coercion resistance.

Credential signing. All TRIP credentials include a signature of the kiosk that produced them, not shown in Fig. 3. This signature ensures that credentials are authorized and traceable to a particular registrar and check-in event, and also counters subtle attacks against related prior approaches [132] in which a coercer identifies a voter's real credential by forging a mathematically-related fake credential [20, 41, 134].

4.6 Optional extensions to the base design

The design of Votegral supports a few optional extensions, which might be included or excluded for policy reasons.

Voting History. Voters can save and view their voting history on their device to further enhance individual verifiability, as discussed in Appendix M.1. Coercion resistance remains intact because voting with a fake credential fabricates a fake voting history.

Reducing Credential Exposure. Although the protocol above ensures that theft of and voting with a credential's secret is eventually detectable, an extension in Appendix M.2 reduces the "window of vulnerability" by ensuring that credential theft is more promptly detectable *by activation time*.

Resisting Extreme Coercion. A few voters might face extreme coercion, such as by being searched immediately after registration. An extension in Appendix M.3 allows voters to delegate their vote within the privacy booth, e.g., to a political party, leaving the booth holding only fake credentials.

4.7 Limitations

Like any system, Votegral has important limitations. Votegral assumes the public roster of eligible voters is correct, and cannot address the suppression or fakery of voters on this roster, as discussed in Appendix C. Systematically addressing other threat vectors, such as voter impersonation (Appendix F), side channels (Appendix N), and rare "Ramanujan voters" (Appendix E.3), are also beyond the scope of this paper.

5 Informal Security Analysis

A voting system is secure if it resists attacks that manipulate the election outcome, while acknowledging that all participants in the system are untrustworthy to some extent. This section shows how Votegral achieves end-to-end verifiability and coercion resistance, resisting two of the three adversaries presented in Appendix I.2: integrity and coercion.

We do not analyze Votegral's resistance to the privacy adversary here, as this paper focuses primarily on verifiability and coercion resistance. Generally, Votegral ensures privacy because decryption of a voter's real ballot requires compromising all election authority members, which is not possible per our threat model. We present a proof sketch in Appendix L.2.

5.1 Individual and universal verifiability

The integrity adversary's goal is to manipulate the election outcome *without detection*. By achieving end-to-end verifiability via making each step of the election process verifiable, Votegral prevents this adversary from achieving this goal.

Individual Verifiability. This notion of verifiability ensures that voters can verify that their submitted real ballot reflects their intended vote and is included in the final tally. Since this paper focuses on the registration process (TRIP), our security analysis centers on achieving the latter: ensuring that the cast ballot counts.[1] To achieve this in a system with real and fake credentials, the voter must be convinced that they have obtained their real credential. The integrity adversary can attempt to steal the voter's real credential by either impersonating the voter during registration or by issuing the voter a fake credential and claiming it as their real one.

To prevent impersonation, Votegral publishes the real credential's public component c_{pc} alongside the voter's identity on the ledger at check-out. This enables the voter's device to detect unauthorized registration sessions under the voter and alert them of this new registration event. If the voter denies the event, the adversary is detected, and the voter can take remedial actions, such as re-registering to replace c_{pc}, thereby invalidating the attacker's registration session.

To prevent an adversary from claiming a fake credential as real, voters are educated through an instructional video on the four-step process for creating a real credential and how it differs from a fake one. In §7.5, we present voter performance in this process from our user study. Consequently, the integrity adversary's only chance of success is to guess the envelope—the ZKP challenge—that the voter selects. While the adversary can control the number of envelopes in the booth, and create duplicate envelopes (envelopes with identical challenges), they cannot influence the voter's actions, such as envelope selection or the number of envelopes consumed (each credential consumes one envelope). This makes the adversary's success probability minimal, and it becomes negligible over repeated attacks against many voters, usually necessary to influence an election. Appendix L.3 presents formal proofs of individual verifiability supporting this claim, along with a more-detailed informal security analysis.

Universal Verifiability. The second verifiability notion of *universal verifiability* enables anyone to verify the election results by ensuring the integrity of the tallying process. Since this paper focuses on TRIP, Votegral's voting and tallying schemes align closely with those presented by JCJ. For related security proofs, we refer to the JCJ paper [78].

5.2 Coercion resistance

The coercion adversary aims to influence election outcomes by pressuring voters either to cast a specific vote, or not to vote. Coercion resistance ensures this adversary cannot determine whether a voter complied, thereby thwarting their objective. This section summarizes how Votegral achieves coercion resistance, with formal proofs available in Appendix L.1.

Votegral achieves coercion resistance by enabling voters to create fake credentials, allowing them to appear compliant while concealing their real credential. These fake credentials are cryptographically and visibly indistinguishable from real ones, as discussed in §4.3 on credential issuance. This indistinguishability also holds when a voter uses a fake credential to cast fake votes. Thus, the coercer cannot determine voter's compliance based on credential appearance or behavior.

Since the coercer cannot observe the voter creating their credential—the only way to distinguish these credentials—the coercer may attempt, before registration, to demand the voter to create a specific number of fake credentials and present them along with one additional credential—their real one. However, voters can always generate one more fake credential, maintaining the secrecy of their real credential.

Unable to discern compliance through voter actions, the coercer might turn to the ledger to determine whether the voter complied with their demands. The ledger reveals the voter's associated public component c_{pc}, the total number of envelopes used in the system and the final tally. Regarding c_{pc}, the coercer might attempt to encrypt the credentials' c_{pk} given by the voter with the election authority's public key. Encryptions are cryptographically randomized, however, so even if the adversary possesses the real credential, the resulting c_{pc} will differ from the one on the ledger. Similarly, decrypting c_{pc} is infeasible as the adversary cannot compromise all the tally servers, as per the threat model, and thus cannot reconstruct the tally service's private key. As for the ledger's disclosure, in aggregate, of the number of envelopes used and the tally, coercion remains ineffective due to statistical uncertainty stemming from the actions of voters the adversary does not control. For example, if the tally shows 5 votes each for candidates A and B, a coercer demanding

[1] Numerous works [25, 42] have been proposed to help voters ensure that the vote they cast contains their intended vote.

a vote for A cannot determine whether the coerced voter contributed to A's votes or voted for B alongside others.

6 Implementation

Our full Votegral prototype consists of 9,182 lines of code as counted by CLOC [48], broken down further in Appendix O.

TRIP is 2,633 lines of Go [98], and uses Kyber [89] for cryptographic operations, gozxing [69] for QR code processing, gofpdf [79] and pdfcpu [13] for printing and reading QR codes. We use Schnorr signatures and SHA-256 for the signing and hashing algorithms, respectively. For both signatures and encryption, we use the Ed25519 elliptic curve.

The rest of Votegral comprises 1,816 lines of Go, utilizing Kyber [89] for cryptographic operations, Bayer Groth [23] for shuffling ElGamal ciphertexts, and a distributed deterministic tagging protocol [136]. Votegral uses a C implementation of Bayer Groth for shuffle proof generation and verification [47]. Unlike a complete system, Votegral simulates each phase of an e-voting system, focusing on the cryptographic operations as they incur the highest computational costs.

7 Experimental Evaluation

To evaluate whether TRIP represents a practical approach to registration for coercion-resistant online voting, we examine the following key questions:

- Q1: Is TRIP fast enough in practice to accommodate voters who typically spend only a few minutes in a booth?
- Q2: How does TRIP's registration-phase performance compare to that of state-of-the-art e-voting systems?
- Q3: How does an online voting system using TRIP perform, and scale, measuring the complete "end-to-end" (E2E) voting pipeline from setup though tallying?
- Q4: Can voters effectively register using TRIP, and protect verifiability by identifying a malfunctioning kiosk?

To compare the computational cost of Votegral and answer Q2 and Q3, we compare TRIP against three e-voting protocols: (1) Swiss Post [8]—a verifiable secret ballot system used in Switzerland—(2) VoteAgain [96]—a coercion-resistant voting system based on deniable re-voting–and (3) Civitas [41]—a verifiable and coercion-resistant voting system. We include Swiss Post's voting system as it represents the state-of-the-art in verifiable secret-ballot systems, despite lacking coercion resistance. We chose VoteAgain for its efficient tallying process, despite lacking end-to-end verifiability. We compare against Civitas because it is a well-known coercion-resistant, end-to-end verifiable system based on JCJ [55, 85, 86, 103–105]. We omit coercion-resistant systems that rely on cryptographic primitives still deemed impractical [123], or that were unfortunately not runnable despite our efforts, including Estonia's system based on deniable re-voting [1].

7.1 Experimental setup and benchmarks

For §7.2, we run TRIP across four distinct hardware setups: (L1) a Point-of-Sale Kiosk (Quad-core Cortex-A17, 2GB RAM, Linaro), (L2) a Raspberry Pi 4 (Quad core Cortex-A72, 4GB RAM, Raspberry Pi), (H1) a Parallels VM on Macbook Pro (M1 Max 4 cores, 8GB RAM, Ubuntu 22.04.2), and (H2) a Beelink GTR7 (AMD Ryzen 7840HS, 32GB RAM, Ubuntu 22.04.4). (L) devices represent resource-constrained setups.

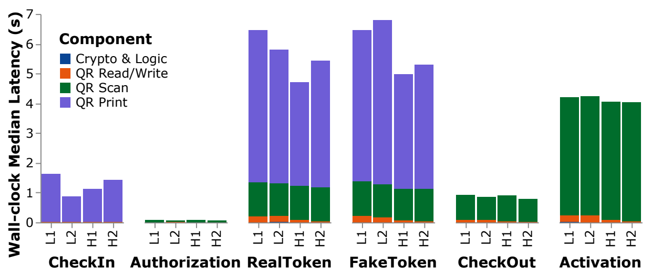
The Macbook Pro serves as our baseline, while the Raspberry Pi and Beelink are compact computational units suitable for integration into devices with QR printing and scanning capabilities. The Point-of-Sale Kiosk, used in our user study [97], include both computational elements and QR peripherals (receipt printer and barcode/QR scanner). However, due to issues with the kiosk's built-in receipt printer, particularly with tearing receipts, we replaced it with a dedicated EPSON TM-T20III printer featuring an automatic cutter for easier receipt handling. To ensure consistency across all devices, we equipped each configuration with this printer and a Bluetooth-connected barcode/QR scanner, as wired connections were not compatible across all four configurations.

In §7.4, our end-to-end evaluation simulates the main phases of an election—Registration, Voting and Tally—focusing on the computational elements. This focus is because cryptographic primitives usually incur the most expensive computational costs, making it a common metric for comparing different e-voting systems. For these experiments, we use Deterlab [28] featuring a bare-metal node with two AMD EPYC 7702 processors for a total of 128 CPU cores (256 threads) and 256GB RAM. The latency of the Swiss Post system represents a realistic benchmark, as it is a government-approved e-voting system deployed in Switzerland.

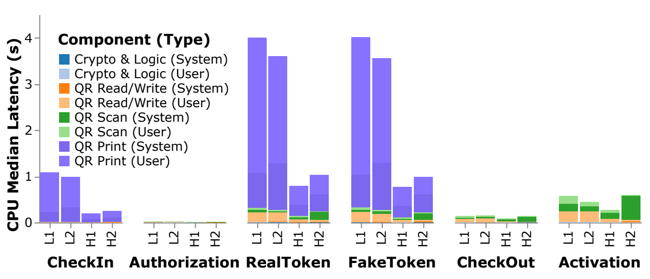
7.2 Voter-observable registration latencies

This experiment addresses Q1: whether TRIP's performance is adequate for realistic use in environments where voters typically spend a few minutes each in the privacy booth. This experiment also explores whether TRIP can operate on low-cost, resource-constrained, or battery-powered devices.

We scripted TRIP to issue one real and one fake credential without human involvement, measuring all user-observable wall-clock delays across all registration phases, including cryptographic operations, QR code scanning ("QR Scan"), encoding/decoding ("QR Read/Write"), and receipt printing ("QR Print"). Since QR printing and scanning are mechanical components, they introduce unique challenges in latency measurement. For QR printing, we adapted CUPS [12] to capture CPU and wall-clock latency from job initiation to completion. For QR scanning, we measured the time when TRIP starts to receive data to when it captures the terminating character. We ran TRIP for 10 consecutive registrations on each hardware, recording wall-clock and CPU latencies for each registration component and phase (Figs. 4a and 4b).

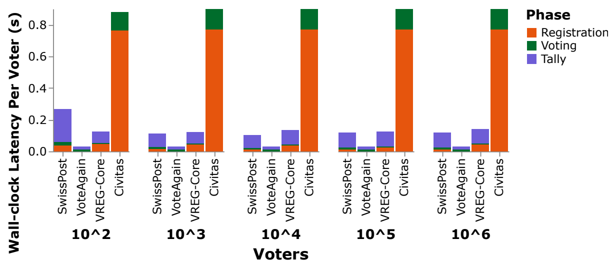


(a) Wall-clock Time incurred by each sub task in TRIP

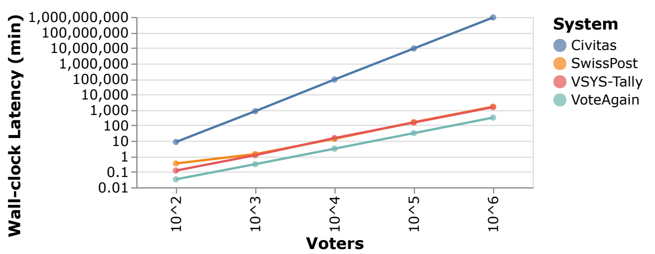


(b) CPU Time incurred by each sub task of TRIP

Figure 4. Execution latency of TRIP: (L) represents resource-constrained devices while (H) represents resource-abundant devices.



(a) Latency per voter for registration, voting, and tallying phases. We exclude the Civitas tally phase to avoid skewing the graph's scale.



(b) Tally Phase latency

Figure 5. Comparing Phase Execution across Voting Systems. Swiss Post is an end-to-end verifiable voting system. VoteAgain is a coercion-resistant system based on deniable re-voting. Civitas implements the JCJ voting system, offering both verifiability and coercion-resistance. Due to Civitas' quadratic time complexity, we extrapolate its latency beyond 10^4 voters. For each voting system, we used four election authority members. With VoteAgain, since this was not a configurable parameter, we ran the mixing process four times.

First, the maximum wall-clock latency perceived by voters during registration is under 19.7 seconds on all deployments, with the longest wall-clock latency for any specific registration process being under 6.5 seconds. In in-person voting, for comparison, the duration that a voter stays inside a booth varies greatly, depending on the complexity of the ballot. A voting time estimator tool [3] states that marking a single-race ballot—where a voter chooses one candidate from many—takes between 19 and 44 seconds (90% confidence interval). Assuming voters instantly execute the mechanical tasks, QR printing and scanning, TRIP meets the lower end of this time range, even on resource-constrained devices.

Second, QR-related tasks significantly contribute to the total wall-clock latency: QR Printing and scanning account for at least 69.5% of the time, with a median overhead of 97.5%. In particular, it takes about 948 milliseconds on average to scan each QR code across devices. This lengthy duration is primarily due to QR data (13-356 bytes) transfer times from the scanner to the computational devices. We expect that a more suitable barcode/QR scanner would be in use, such as the kiosk's embedded scanner, which scans even 300-byte QR codes almost instantly. Therefore, using a more efficient scanner could reduce the overall registration latency by about a third, on average, as each registration run currently takes 7 seconds, on average, to scan QR codes.

Third, we find that resource-constrained devices (L1 and L2) perform up to 19.8% slower than higher-end devices (H1 and H2). Specifically, L1 is the slowest with a wall-clock latency of 19.7 seconds, while H1 is the fastest at 15.8 seconds. This difference can be attributed to the CPU time distribution shown in Fig. 4b. On average, the CPU latency for resource-constrained devices is 260% higher than that for higher-end devices. In particular, QR printing on these devices takes 380% higher than that on higher-end devices. Despite these significant increases in CPU latency, the overall wall-clock latency only rises by an average of 16.5%.

In answer to Q1, we find that TRIP appears suitable for environments on time scales comparable to those applicable to in-person voting, even on resource-constrained devices.

7.3 Registration performance across systems

We now examine Q2: How does registration performance in TRIP compare to the registration-time costs of other e-voting systems? This experiment focuses on computational costs, excluding the I/O-related latencies such as QR code printing and scanning considered earlier for Q1. We focus on computation cost for two reasons: TRIP's I/O-bound phases lack direct analogs in existing systems, making "apples-to-apples" comparison impractical; and second, computation cost is a key metric in e-voting research, as cryptographic operations, such as shuffling, decryption, and zero-knowledge proof generation and verification, typically dominate and can be prohibitive for deployment. We compare the cryptographic performance between TRIP, Swiss Post, VoteAgain and Civitas while varying the number of voters.

For TRIP, which includes voter-interaction features like QR codes, we use a configuration termed "TRIP-Core" that omits all QR-related tasks to isolate the cryptographic operations. Swiss Post provided us their end-to-end election simulator, enabling us to run their cryptographic functions. VoteAgain and Civitas did not require any changes from the code provided by their respective papers. Figure 5a presents our per-voter results, which also include data for voting and tallying phases; these aspects are discussed further in §7.4.

In the experimental configuration for 1 million voters, the registration latency per voter for TRIP, Swiss Post, VoteAgain, and Civitas is 45 ms, 13 ms, 0.1 ms, and 771 ms, respectively (Fig. 5a). VoteAgain significantly outperforms all other systems due to its lack of verifiability, reducing the process to generating a key pair for each voter. TRIP is 246% slower than Swiss Post but 94% faster than Civitas. The difference in latency between TRIP and Civitas partly results from Civitas using RSA while TRIP, VoteAgain and Swiss Post employ more-efficient elliptic curves. The difference between TRIP and Swiss Post reflects the additional overhead needed to provide coercion-resistance. Nevertheless, these results suggest that TRIP lies in the same general performance range as already-deployed e-voting systems.

7.4 End-to-end performance across systems

To address Q3, we broaden our perspective to encompass the "end-to-end" operation of an e-voting system by comparing Votegral against Swiss Post, VoteAgain and Civitas. As part of Votegral, we used TRIP-Core as the registration component, and the voting and tallying schemes detailed in Appendix K. We measured the execution latency of each phase for configurations ranging from 100 to 1 million voters while maintaining four talliers. Because tallying in Civitas has quadratic complexity, we extrapolated its results after 1,000 voters due to time constraints. Figure 5a depicts the total execution latency per voter across each phase (excluding Civitas's tally phase), and Figure 5b compares systems' tally latencies.

We observe that during the voting phase alone, the latency per voter for TRIP, Swiss Post, VoteAgain and Civitas is 5ms, 10ms, 1ms, 128ms, respectively. The latency cost of voting is independent of voting population in all systems, unsurprisingly since this phase is "embarrassingly parallel." Voting in Votegral performs twice as fast as Swiss Post and 24 times faster than Civitas but 5 times slower than VoteAgain.

In the tally phase (Fig. 5b), Votegral requires approximately 25 hours for 1 million voters, compared to 19 hours for VoteAgain, 27 hours for Swiss Post and an impractical 1,768 *years* (estimated) for Civitas. While Civitas can enhance its tally performance by dividing ballots into groups, this method weakens coercion resistance. While Swiss Post's tally phase is 7% slower compared to Votegral, Fig. 5a shows that the speed gained by TRIP in the voting and tally phases are offset by its slower registration latency compared to Swiss Post.

In summary, we find that Votegral significantly outperforms Civitas in "end-to-end" performance across the pipeline, while also competing closely with the deployed but non-coercion-resistant Swiss Post voting system and the coercion-resistant but non-verifiable VoteAgain system.

7.5 Usability studies of TRIP

To address Q4, whether voters can use TRIP and detect a compromised kiosk, we conducted a usability study with 150 participants in a U.S. Northeastern city. Our findings, detailed in our companion paper [97], are summarized below. Our studies were approved by our institutional review board.

We initially conducted two preliminary user studies involving 77 participants to gain insights on our design.[2] This led to two enhancements in TRIP. First, we replaced a QR code on the check-in ticket with a barcode, as participants often mistook it for the kiosk's first printed QR code used to create real credentials. Second, to remind participants to pick and scan an envelope after the first QR, we added a matching symbol on the receipt and envelopes, prompting the voter to select an envelope bearing the same symbol.

In our main study of 150 participants, TRIP achieved an 83% success rate and a System Usability Scale score of 70.4 – slightly above the industry average of 68. Participants found TRIP just as usable as a simplified one involving only real credentials. Furthermore, 47% of participants who received security education could detect and report a misbehaving kiosk, while 10% could do so without security education. Assuming each voter has a 10% chance of detecting and reporting a malicious kiosk, the probability that a compromised kiosk could trick 50 voters without detection is under 1%. For 1000 voters, that probability drops to a cryptographically negligible $1/2^{152}$. More details on this study is in Appendix P.

8 Related Work

In JCJ [78], the registrar issues a real credential via an abstract untappable channel. The registrar and voter are then assumed to protect the secrecy of this real credential. TRIP implements this untappable channel using a paper-based workflow utilizing interactive zero-knowledge proofs.

Civitas [41] proposes that voters interact with multiple registration tellers to reduce trust in any one teller. Asking real-world voters to interact with several tellers, however, incurs significant complexity and convenience costs, whose

[2]The first preliminary study involved 41 PhD students, the second involved 36 individuals from varying locations across this northeastern city.

practicality has never been tested with a usability study. Further, the election authority faces the unenviable prospect of justifying this convenience cost to voters that risks sounding like: "You must interact with several registration tellers because you can't fully trust any of them, even though we hired and trained them and they answer to us." Election authorities usually want to and are even mandated to promote trust in elections and electoral processes, not to undermine that very trust! In TRIP, coerced voters avoid such a predicament: they know they must either trust the kiosk (and realistically the registrar) not to collude with their coercer, or else comply with the coercer's demands—a simpler binary choice.

In Krivoruchko's work on registration [87], voters must generate their real credential, encrypt it, and then provide the encrypted version to the registrar. This process ensures that the registrar never obtains the real credential, eliminating the need for the registrar to prove the credential's integrity. Nevertheless, this approach has two shortcomings with respect to coercion resistance: First, voters must possess a device prior to registration, which the coercer could have compromised or confiscated beforehand. Second, the scheme lacks a mechanism that proves to the registrar that the voter's device knows the real credential from the encrypted version it submits to the registrar. This is essential to prevent voters from giving the registrar with an encrypted version of a real credential that was generated by a coercer, thus rendering the credential inaccessible to the voter.

Prior work has used interactive zero-knowledge proofs (IZKP) for receipt-free, voter-verifiable voting with Direct Recording Electronic (DRE) machines [38, 99]. In particular, Moran and Noar's approach [31] relied on DRE machines to place an opaque shield over part of the receipt and asked voters to enter random words as a challenge. In contrast, TRIP uses IZKPs for voter registration and simplifies the process for users with a design that involves selecting an envelope and scanning its QR code. As discussed in Appendix E.3, we do not deem it necessary to shield the printed commitment from the voter: it is hard for ordinary users to interpret QR codes or compute cryptographic functions without electronic devices, a much easier unauthorized use of which is simple recording (Appendix E.1).

9 Conclusion

This paper presents Votegral, a coercion-resistant, verifiable, user-studied online voting system based on fake credentials, requiring no trusted device in credential issuance. We find that Votegral is competitive in performance and efficiency with state-of-the-art e-voting systems. Formal security proofs confirm that TRIP provides individual verifiability and coercion resistance. Finally, usability studies offer evidence that Votegral is understandable and usable by ordinary voters.

10 Acknowledgments

The authors would like to thank the E-Voting Group at Bern University of Applied Sciences, namely Rolf Haenni and Eric Dubuis, for their valuable feedback. This project was supported in part by the Fulbright U.S. Student Program, the Swiss Government Excellence Scholarships for Foreign Scholars, the armasuisse Science and Technology, the AXA Research Fund, and the US ONR grant N000141912361.

References

[1] [n. d.]. e-Democracy & open data - e-Estonia. https://e-estonia.com/solutions/e-governance/e-democracy/ https://e-estonia.com/solutions/e-governance/e-democracy/.

[2] [n. d.]. E-voting and e-elections. https://www.id.ee/en/article/e-voting-and-e-elections/ https://www.id.ee/en/article/e-voting-and-e-elections/.

[3] [n. d.]. Voting Time Estimator. https://go41.com/vote/estimator.htm https://go41.com/vote/estimator.htm.

[4] 1989. Election Code (10 ILCS 5/). https://www.ilga.gov/legislation/ilcs/ilcs5.asp?ActID=170&ChapterID=3.

[5] 2009. Military and Overseas Voter Empowerment "MOVE" Act. Pub. L. No. 111-84, §§ 577-83(a). Enacted as part of the National Defense Authorization Act for Fiscal Year 2010.

[6] 2018. *Securing the Vote: Protecting American Democracy.* National Academies Press, Washington, D.C. https://www.nap.edu/catalog/25120

[7] 2021. District of New Jersey | Postal Employee Admits Dumping Mail, Including Election Ballots Sent to West Orange Residents | United States Department of Justice. https://www.justice.gov/usao-nj/pr/postal-employee-admits-dumping-mail-including-election-ballots-sent-west-orange-residents

[8] 2021. *Protocol of the Swiss Post Voting System: Computational Proof of Complete Verifiability and Privacy.* Technical Report 1.0.0. https://gitlab.com/swisspost-evoting/e-voting/e-voting-documentation

[9] 2022. Four people plead guilty in North Carolina ballot probe of 2016 and 2018 elections. *NBC News* (Sept. 2022). https://www.nbcnews.com/politics/elections/four-people-plead-guilty-north-carolina-ballot-probe-2016-2018-electio-rcna49534

[10] 2024. ID-card | e-estonia. https://e-estonia.com/solutions/estonian-e-identity/id-card/ https://e-estonia.com/solutions/estonian-e-identity/id-card/.

[11] 2024. Moldovan Police Accuse Pro-Russian Oligarch Of $39M Vote-Buying Scheme. *Radio Free Europe/Radio Liberty* (Oct. 2024). https://www.rferl.org/a/moldova-police-accuse-shor-russia-oligarch-39m-vote-buying/33172951.html

[12] 2024. OpenPrinting/cups. https://github.com/OpenPrinting/cups https://github.com/OpenPrinting/cups.

[13] 2024. pdfcpu/pdfcpu. https://github.com/pdfcpu/pdfcpu https://github.com/pdfcpu/pdfcpu.

[14] 2024. US election: Bomb threats and printer glitches disrupt voting in battleground states. *Sky News* (Nov. 2024). https://news.sky.com/story/us-election-bomb-threats-and-printer-glitches-disrupt-voting-in-key-swing-states-13249161

[15] Dirk Achenbach, Carmen Kempka, Bernhard Löwe, and Jörn Müller-Quade. 2015. Improved Coercion-Resistant Electronic Elections through Deniable Re-Voting. *USENIX Journal of Election Technology and Systems (JETS)* (Aug. 2015). https://www.usenix.org/conference/jets15/workshop-program/presentation/achenbach

[16] Ben Adida. 2008. Helios: Web-based Open-Audit Voting.. In *USENIX Security Symposium*, Vol. 17. 335–348.

[17] Steven Adler, Zoë Hitzig, Shrey Jain, Catherine Brewer, Wayne Chang, Renée DiResta, Eddy Lazzarin, Sean McGregor, Wendy Seltzer, Divya

Siddarth, Nouran Soliman, Tobin South, Connor Spelliscy, Manu Sporny, Varya Srivastava, John Bailey, Brian Christian, Andrew Critch, Ronnie Falcon, Heather Flanagan, Kim Hamilton Duffy, Eric Ho, Claire R. Leibowicz, Srikanth Nadhamuni, Alan Z. Rozenshtein, David Schnurr, Evan Shapiro, Lacey Strahm, Andrew Trask, Zoe Weinberg, Cedric Whitney, and Tom Zick. 2025. Personhood credentials: Artificial intelligence and the value of privacy-preserving tools to distinguish who is real online. https://doi.org/10.48550/arXiv.2408.07892

[18] Makhabbat Amanbay. 2023. The Ethics of AI-generated Art. https://papers.ssrn.com/abstract=4551467

[19] Andrew W. Appel, Richard A. DeMillo, and Philip B. Stark. 2020. Ballot-Marking Devices Cannot Ensure the Will of the Voters. *Election Law Journal: Rules, Politics, and Policy* 19, 3 (Sept. 2020), 432–450. https://doi.org/10.1089/elj.2019.0619

[20] Roberto Araújo, Sébastien Foulle, and Jacques Traoré. 2010. A Practical and Secure Coercion-Resistant Scheme for Internet Voting. In *Towards Trustworthy Elections: New Directions in Electronic Voting*, David Chaum, Markus Jakobsson, Ronald L. Rivest, Peter Y. A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida (Eds.). Springer, Berlin, Heidelberg, 330–342. https://doi.org/10.1007/978-3-642-12980-3_20

[21] James Austgen, Andrés Fábrega, Sarah Allen, Kushal Babel, Mahimna Kelkar, and Ari Juels. 2023. DAO Decentralization: Voting-Bloc Entropy, Bribery, and Dark DAOs. https://doi.org/10.48550/arXiv.2311.03530

[22] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of Usability Studies* 4, 3 (2009), 114–123.

[23] Stephanie Bayer and Jens Groth. 2012. Efficient Zero-Knowledge Argument for Correctness of a Shuffle. In *Advances in Cryptology – EUROCRYPT 2012*, David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, David Pointcheval, and Thomas Johansson (Eds.). Vol. 7237. Springer Berlin Heidelberg, Berlin, Heidelberg, 263–280. http://link.springer.com/10.1007/978-3-642-29011-4_17

[24] Josh Benaloh. 1987. *Verifiable Secret-Ballot Elections*. Ph.D. Thesis. https://www.microsoft.com/en-us/research/publication/verifiable-secret-ballot-elections/

[25] Josh Benaloh. 2007. Ballot Casting Assurance via Voter-Initiated Poll Station Auditing. In *2007 USENIX/ACCURATE Electronic Voting Technology Workshop*. Boston, MA. https://www.usenix.org/conference/evt-07/ballot-casting-assurance-voter-initiated-poll-station-auditing

[26] Josh Benaloh. 2013. Rethinking Voter Coercion: The Realities Imposed by Technology. *USENIX Journal of Election Technology and Systems* 1, 1 (2013).

[27] Josh Benaloh and Dwight Tuinstra. 1994. Receipt-free secret-ballot elections. In *Proceedings of the 26th annual ACM symposium on Theory of Computing (STOC '94)*. 544–553. https://doi.org/10.1145/195058.195407

[28] Terry Benzel. 2011. The science of cyber security experimentation: the DETER project. In *Proceedings of the 27th Annual Computer Security Applications Conference (ACSAC '11)*. Association for Computing Machinery, New York, NY, USA, 137–148. https://doi.org/10.1145/2076732.2076752

[29] David Bernhard, Véronique Cortier, Pierrick Gaudry, Mathieu Turuani, and Bogdan Warinschi. 2018. Verifiability Analysis of CHVote. https://eprint.iacr.org/undefined/undefined

[30] Matthew Bernhard. 2021. Risk-limiting audits: A practical systematization of knowledge. In *E-Vote-ID 2021*. 162. https://mbernhard.com/papers/RLA_Threat_Model.pdf

[31] Matthew Bernhard, Allison McDonald, Henry Meng, Jensen Hwa, Nakul Bajaj, Kevin Chang, and J. Alex Halderman. 2020. Can Voters Detect Malicious Manipulation of Ballot Marking Devices?. In *2020 IEEE Symposium on Security and Privacy*. 679–694. https://doi.org/10.1109/SP40000.2020.00118

[32] Christian Blum and Christina Isabel Zuber. 2016. Liquid Democracy: Potentials, Problems, and Perspectives. *The Journal of Political Philosophy* 24, 2 (June 2016), 162–182.

[33] Joseph Bonneau, Elie Bursztein, Ilan Caron, Rob Jackson, and Mike Williamson. 2015. Secrets, Lies, and Account Recovery: Lessons from the Use of Personal Knowledge Questions at Google. In *Proceedings of the 24th international conference on World Wide Web*. 141–150.

[34] Maria Borge, Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, and Bryan Ford. 2017. Proof-of-Personhood: Redemocratizing Permissionless Cryptocurrencies. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. 23–26. https://doi.org/10.1109/EuroSPW.2017.46

[35] John Brooke. 1996. SUS: A 'Quick and Dirty' Usability Scale. In *Usability Evaluation In Industry*. CRC Press.

[36] Eleno Castro and Randy Kotti. 2022. *Saving Democracy: Reducing Gang Influence on Political Elections in El Salvador*. Master's thesis.

[37] Suzanne Chapman, Christopher Patten, and Roberto Spreggiaro. 2024. *A Review of the literature on voter verification and ballot review*. Technical Report NIST GCR 24-052. National Institute of Standards and Technology (U.S.), Gaithersburg, MD. NIST GCR 24–052 pages. https://nvlpubs.nist.gov/nistpubs/gcr/2024/NIST.GCR.24-052.pdf

[38] David Chaum. 2004. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security & Privacy* 2, 1 (Jan. 2004), 38–47. https://doi.org/10.1109/MSECP.2004.1264852

[39] David Chaum and Torben Pryds Pedersen. 1993. Wallet Databases with Observers. In *Advances in Cryptology — CRYPTO' 92*. Berlin, Heidelberg, 89–105. https://doi.org/10.1007/3-540-48071-4_7

[40] Bobby Chesney and Danielle Citron. 2019. Deep Fakes: A Looming Challenge for Privacy. *California Law Review* 107, 6 (2019). https://doi.org/10.15779/Z38RV0D15J

[41] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. 2008. Civitas: Toward a Secure Voting System. In *2008 IEEE Symposium on Security and Privacy*. 354–368. https://doi.org/10.1109/SP.2008.32

[42] Véronique Cortier, Alexandre Debant, Pierrick Gaudry, and Stéphane Glondu. 2024. Belenios with Cast as Intended. In *Financial Cryptography and Data Security. FC 2023 International Workshops (Lecture Notes in Computer Science)*, Aleksander Essex, Shin'ichiro Matsuo, Oksana Kulyk, Lewis Gudgeon, Ariah Klages-Mundt, Daniel Perez, Sam Werner, Andrea Bracciali, and Geoff Goodell (Eds.). Springer Nature Switzerland, Cham, 3–18. https://doi.org/10.1007/978-3-031-48806-1_1

[43] Véronique Cortier, Jannik Dreier, Pierrick Gaudry, and Mathieu Turuani. 2019. A simple alternative to Benaloh challenge for the cast-as-intended property in Helios/Belenios. (2019). https://hal.inria.fr/hal-02346420

[44] Malcolm Crook and Tom Crook. 2011. Reforming Voting Practices In a Global Age: The Making and Remaking of the Modern Secret Ballot in Britain, France and the United States c. 1600—c. 1950. *Past & Present* 212 (2011), 199–237. https://www.jstor.org/stable/23014789

[45] Scott A. Crosby and Dan S. Wallach. 2009. Efficient data structures for tamper-evident logging. In *Proceedings of the 18th conference on USENIX security symposium (SSYM'09)*. USENIX Association, USA, 317–334.

[46] Scott A. Crosby and Dan S. Wallach. 2009. Efficient Data Structures for Tamper-Evident Logging. In *USENIX Security Symposium* (Montreal, Canada).

[47] Anders Dalskov. 2021. anderspkd/groth-shuffle: Bayer-Groth shuffle. https://github.com/anderspkd/groth-shuffle/tree/master

[48] Al Danial. [n. d.]. Counting Lines of Code. http://cloc.sourceforge.net/.

[49] Jerusalem Demsas. 2020. Can you take a photo of your ballot? Ballot selfie rules in every state, explained | Vox. *Vox* (Oct. 2020). https://www.vox.com/21523858/ballot-selfies-state-rules

[50] Roger Dingledine and Paul Syverson. 2003. Reliable MIX Cascade Networks through Reputation. In *Financial Cryptography*, Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, and Matt Blaze (Eds.). Vol. 2357. Springer Berlin Heidelberg, Berlin, Heidelberg, 253–268. http://link.springer.com/10.1007/3-540-36504-4_18

[51] Verena Distler, Marie-Laure Zollinger, Carine Lallemand, Peter B. Roenne, Peter Y. A. Ryan, and Vincent Koenig. 2019. Security - Visible, Yet Unseen?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300835

[52] John R. Douceur. 2002. The Sybil Attack. In *Peer-to-Peer Systems*, Peter Druschel, Frans Kaashoek, and Antony Rowstron (Eds.). Springer, Berlin, Heidelberg, 251–260. https://doi.org/10.1007/3-540-45748-8_24

[53] Piret Ehin, Mihkel Solvak, Jan Willemson, and Priit Vinkel. 2022. Internet voting in Estonia 2005–2019: Evidence from eleven elections. *Government Information Quarterly* 39, 4 (Oct. 2022), 101718. https://doi.org/10.1016/j.giq.2022.101718

[54] Robert Epstein and Ronald E. Robertson. 2015. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. *Proceedings of the National Academy of Sciences* 112, 33 (Aug. 2015), E4512–E4521. https://doi.org/10.1073/pnas.1419828112

[55] Ehsan Estaji, Thomas Haines, Kristian Gjøsteen, Peter B. Rønne, Peter Y. A. Ryan, and Najmeh Soroush. 2020. Revisiting Practical and Usable Coercion-Resistant Remote E-Voting. In *E-Vote-ID 2020: Electronic Voting*. 50–66. https://doi.org/10.1007/978-3-030-60347-2_4

[56] Christopher Famighetti. 2016. Long voting lines: Explained. *Brennan Center for Justice* (2016). https://www.brennancenter.org/sites/default/files/analysis/Long_Voting_Lines_Explained.pdf

[57] Robert Faulkner. 2022. What Is TEMPEST? | Cybersecurity RF & Electromagnetic Shielding. https://blog.enconnex.com/what-is-tempest

[58] Christian Feier, Stephan Neumann, and Melanie Volkamer. 2014. Coercion-Resistant Internet Voting in Practice. In *Informatik 2014*. https://dl.gi.de/handle/20.500.12116/2749

[59] Dragan Filipovich, Miguel Niño-Zarazúa, and Alma Santillán Hernández. 2021. Voter coercion and pro-poor redistribution in rural Mexico. (July 2021). https://doi.org/10.35188/UNU-WIDER/2021/081-8

[60] Caitriona Fitzgerald, Pamela Smith, and Susannah Goodman. 2016. The Secret Ballot At Risk: Recommendations for Protecting Democracy. (Aug. 2016).

[61] James Flamino, Alessandro Galeazzi, Stuart Feldman, Michael W. Macy, Brendan Cross, Zhenkun Zhou, Matteo Serafino, Alexandre Bovet, Hernán A. Makse, and Boleslaw K. Szymanski. 2023. Political polarization of news media and influencers on Twitter in the 2016 and 2020 US presidential elections. *Nature Human Behaviour* 7, 6 (June 2023), 904–916. https://doi.org/10.1038/s41562-023-01550-8

[62] Bryan Ford. 2020. A Liquid Perspective on Democratic Choice.

[63] Bryan Ford. 2020. Identity and Personhood in Digital Democracy: Evaluating Inclusion, Equality, Security, and Privacy in Pseudonym Parties and Other Proofs of Personhood. https://doi.org/10.48550/arXiv.2011.02412

[64] Bryan Ford and Jacob Strauss. 2008. An offline foundation for online accountable pseudonyms. In *Proceedings of the 1st Workshop on Social Network Systems (SocialNets '08)*. Association for Computing Machinery, New York, NY, USA, 31–36. https://doi.org/10.1145/1435497.1435503

[65] David Froelicher, Patricia Egger, João Sá Sousa, Jean Louis Raisaro, Zhicong Huang, Christian Vincent Mouchet, Bryan Ford, and Jean-Pierre Hubaux. 2017. UnLynx: A Decentralized System for Privacy-Conscious Data Sharing. In *Proceedings on Privacy Enhancing Technologies*, Vol. 4. 152–170. https://petsymposium.org/2017/papers/issue4/paper54-2017-4-source.pdf

[66] Timothy Frye, Ora John Reuter, and David Szakonyi. 2019. Hitting Them With Carrots: Voter Intimidation and Vote Buying in Russia. *British Journal of Political Science* 49, 3 (July 2019), 857–881. https://doi.org/10.1017/S0007123416000752

[67] Xianyi Gao, Yulong Yang, Can Liu, Christos Mitropoulos, Janne Lindqvist, and Antti Oulasvirta. 2018. Forgetting of Passwords: Ecological Theory and Data. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, Baltimore, MD, 221–238. https://www.usenix.org/conference/usenixsecurity18/presentation/gao-xianyi

[68] Ezequiel Gonzalez-Ocantos, Chad Kiewiet de Jonge, Carlos Meléndez, David Nickerson, and Javier Osorio. 2020. Carrots and sticks: Experimental evidence of vote-buying and voter intimidation in Guatemala. *Journal of Peace Research* 57, 1 (Jan. 2020), 46–61. https://doi.org/10.1177/0022343319884998

[69] gozxing 2021. A Barcode Scanning/Encoding Library for Go.

[70] Michael Graff and Nick Ochsner. 2021. 'This Smacks of Something Gone Awry': A True Tale of Absentee Vote Fraud. https://www.politico.com/news/magazine/2021/11/29/true-tale-absentee-voter-fraud-north-carolina-523238

[71] Jens Groth. 2010. A Verifiable Secret Shuffle of Homomorphic Encryptions. *Journal of Cryptology* 23, 4 (Oct. 2010), 546–579. https://doi.org/10.1007/s00145-010-9067-9

[72] Sufian Hameed, Syed Atyab Hussain, and Sohail Hussain Ali. 2013. SafePass: Authentication under duress for ATM transactions. In *2013 2nd National Conference on Information Assurance (NCIA)*. 1–5. https://doi.org/10.1109/NCIA.2013.6725317

[73] Lydia Hardy. 2020. Voter Suppression Post-Shelby: Impacts and Issues of Voter Purge and Voter ID Laws. *Mercer Law Review* (May 2020).

[74] Landemore Hélène. 2020. *Open Democracy: Reinventing Popular Rule for the Twenty-First Century*. https://press.princeton.edu/books/hardcover/9780691181998/open-democracy

[75] Ashar Javed, David Bletgen, Florian Kohlar, Markus Dürmuth, and Jörg Schwenk. 2014. Secure Fallback Authentication and the Trusted Friend Attack. In *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE, 22–28.

[76] Malcolm Johnson. 2021. Mail Carrier Accused of Throwing Away Mail-in Ballots. *NBC Boston* (Oct. 2021). https://www.nbcboston.com/news/local/mail-carrier-accused-of-throwing-away-mail-in-ballots/2544494/

[77] Ari Juels, Dario Catalano, and Markus Jakobsson. 2005. Coercion-Resistant Electronic Elections (Extended Abstract). In *Workshop on Privacy in the Electronic Society (WPES)* (Alexandria, VA, USA).

[78] Ari Juels, Dario Catalano, and Markus Jakobsson. 2010. Coercion-Resistant Electronic Elections. In *Towards Trustworthy Elections: New Directions in Electronic Voting*. Berlin, Heidelberg, 37–63. https://doi.org/10.1007/978-3-642-12980-3_2

[79] Kurt Jung. 2024. jung-kurt/gofpdf. https://github.com/jung-kurt/gofpdf

[80] Alex Katsomitros. 2025. The great big tech break-up. https://www.worldfinance.com/special-reports/the-great-big-tech-break-up

[81] Fiach Kelly. 2015. Q&A how to vote: No selfies, and leave the badge at home. https://www.irishtimes.com/news/politics/q-a-how-to-vote-no-selfies-and-leave-the-badge-at-home-1.2218688

[82] Habibul Haque Khondker. 2011. Role of the New Media in the Arab Spring. *Globalizations* (Oct. 2011). https://www.tandfonline.com/doi/abs/10.1080/14747731.2011.621287

[83] Christian Killer and Burkhard Stiller. 2019. The Swiss Postal Voting Process and Its System and Security Analysis. In *Electronic Voting*, Robert Krimmer, Melanie Volkamer, Veronique Cortier, Bernhard Beckert, Ralf Küsters, Uwe Serdült, and David Duenas-Cid (Eds.). Springer International Publishing, Cham, 134–149. https://doi.org/10.1007/978-3-030-30625-0_9

[84] Christoph Kling, Jérôme Kunegis, Heinrich Hartmann, Markus Strohmaier, and Steffen Staab. 2015. Voting Behaviour and Power in Online Democracy: A Study of LiquidFeedback in Germany's Pirate Party. *Proceedings of the International AAAI Conference on Web and Social Media* 9, 1 (2015), 208–217. https://doi.org/10.1609/icwsm.v9i1.14618

[85] Reto Koenig, Rolf Haenni, and Stephan Fischli. 2011. Preventing Board Flooding Attacks in Coercion-Resistant Electronic Voting Schemes. In *Future Challenges in Security and Privacy for Academia and Industry*. Berlin, Heidelberg, 116–127. https://doi.org/10.1007/978-3-642-21424-0_10

[86] Kristjan Krips and Jan Willemson. 2019. On Practical Aspects of Coercion-Resistant Remote Voting Systems. In *E-Vote-ID 2019: Electronic Voting*. 216–232. https://doi.org/10.1007/978-3-030-30625-0_14

[87] Taisya Krivoruchko. 2007. Robust Coercion-Resistant Registration for Remote E-voting. In *Proceedings of the IAVoSS Workshop on Trustworthy Elections*.

[88] Oksana Kulyk and Stephan Neumann. 2020. Human Factors in Coercion Resistant Internet Voting–A Review of Existing Solutions and Open Challenges. In *Proceedings of the Fifth International Joint Conference on Electronic Voting*. TalTech press, 189.

[89] kyber 2010 – 2018. The Kyber Cryptography Library.

[90] Ralf Küsters, Julian Liedtke, Johannes Müller, Daniel Rausch, and Andreas Vogt. 2020. Ordinos: A Verifiable Tally-Hiding E-Voting System. In *2020 IEEE European Symposium on Security and Privacy*. https://doi.org/10.1109/EuroSP48549.2020.00022

[91] Antonin Lacelle-Webster and Mark E. Warren. 2021. Citizens' Assemblies and Democracy. In *Oxford Research Encyclopedia of Politics*. Oxford University Press. https://oxfordre.com/politics/display/10.1093/acrefore/9780190228637.001.0001/acrefore-9780190228637-e-1975

[92] Corentin Lavaud, Robin Gerzaguet, Matthieu Gautier, Olivier Berder, Erwan Nogues, and Stephane Molton. 2021. Whispering Devices: A Survey on How Side-channels Lead to Compromised Information. *Journal of Hardware and Systems Security* 5, 2 (June 2021), 143–168. https://doi.org/10.1007/s41635-021-00112-6

[93] J.C.R. Licklider. 1968. The Computer as a Communication Device. *Science and Technology* (April 1968).

[94] Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology* 22 140 (1932), 55–55.

[95] Simon Luechinger, Myra Rosinger, and Alois Stutzer. 2007. The Impact of Postal Voting on Participation: Evidence for Switzerland. *Swiss Political Science Review* 13, 2 (2007), 167–202.

[96] Wouter Lueks, Iñigo Querejeta-Azurmendi, and Carmela Troncoso. 2020. VoteAgain: A scalable coercion-resistant voting system. In *29th USENIX Security Symposium*. 1553–1570. https://www.usenix.org/conference/usenixsecurity20/presentation/lueks

[97] Louis-Henri Merino, Alaleh Azhir, Haoqian Zhang, Simone Colombo, Bernhard Tellenbach, Vero Estrada-Galiñanes, and Bryan Ford. 2024. E-Vote Your Conscience: Perceptions of Coercion and Vote Buying, and the Usability of Fake Credentials in Online Voting. In *2024 IEEE Symposium on Security and Privacy (SP)*. 3478–3496. https://doi.org/10.1109/SP54263.2024.00252

[98] Jeff Meyerson. 2014. The Go Programming language. *IEEE Software* 31, 5 (2014), 104–104.

[99] Tal Moran and Moni Naor. 2006. Receipt-Free Universally-Verifiable Voting with Everlasting Privacy. In *Advances in Cryptology*. https://doi.org/10.1007/11818175_22

[100] National Conference of State Legislatures. 2024. Table 18: States with All-Mail Elections. https://www.ncsl.org/elections-and-campaigns/table-18-states-with-all-mail-elections. Accessed: 2024-12-07.

[101] National Conference of State Legislatures. 2024. Table 3: States with Permanent Absentee Voting Lists. https://www.ncsl.org/elections-and-campaigns/table-3-states-with-permanent-absentee-voting-lists. Accessed: 2024-12-07.

[102] C. Andrew Neff. 2001. A verifiable secret shuffle and its application to e-voting. In *Proceedings of the 8th ACM conference on Computer and Communications Security (CCS '01)*. Association for Computing Machinery, New York, NY, USA, 116–125. https://doi.org/10.1145/501983.502000

[103] André Silva Neto, Matheus Leite, Roberto Araújo, Marcelle Pereira Mota, Nelson Cruz Sampaio Neto, and Jacques Traoré. 2018. Usability Considerations For Coercion-Resistant Election Systems. In *Proceedings of the 17th Brazilian Symposium on Human Factors in Computing Systems (IHC 2018)*. New York, NY, USA, 1–10. https://doi.org/10.1145/3274192.3274232

[104] Stephan Neumann, Christian Feier, Melanie Volkamer, and Reto Koenig. 2013. Towards a practical JCJ/Civitas implementation. In *Informatik angepasst an Mensch, Organisation und Umwelt*.

[105] Stephan Neumann and Melanie Volkamer. 2012. Civitas and the Real World: Problems and Solutions from a Practical Point of View. In *Seventh Conference on Availability, Reliability and Security*. https://doi.org/10.1109/ARES.2012.75

[106] Raymond S. Nickerson. 2002. The production and perception of randomness. *Psychological Review* 109, 2 (April 2002), 330–357. https://doi.org/10.1037/0033-295x.109.2.330

[107] NIST. 2007. *Usability Performance Benchmarks For the Voluntary Voting System Guidelines*. Technical Report.

[108] Puja Ohlhaver, Mikhail Nikulin, and Paula Berman. 2025. Compressed to 0: The Silent Strings of Proof of Personhood. *Stanford Journal of Blockchain Law & Policy* (Jan. 2025). https://stanford-jblp.pubpub.org/pub/compressed-to-0-proof-personhood/release/5

[109] Jon H Pammett and Nicole Goodman. 2013. Consultation and Evaluation Practices in the Implementation of Internet Voting in Canada and Europe.

[110] Sunoo Park, Michael Specter, Neha Narula, and Ronald L Rivest. 2021. Going from bad to worse: from Internet voting to blockchain voting. *Journal of Cybersecurity* 7, 1 (Feb. 2021). https://doi.org/10.1093/cybsec/tyaa025

[111] Richard H. Pildes. 2020. How to Accommodate a Massive Surge in Absentee Voting. *The University of Chicago Law Review* (2020). https://lawreview.uchicago.edu/online-archive/how-accommodate-massive-surge-absentee-voting

[112] Gerald V. Post. 2010. Using re-voting to reduce the threat of coercion in elections. *Electronic Government, an International Journal* 7, 2 (Jan. 2010), 168–182. https://doi.org/10.1504/EG.2010.030926

[113] Swiss Post. 2024. *E-Voting Architecture Document*. Technical Report. v1.4.0.

[114] The Associated Press. 2014. No selfies: Brazil keeps phones out of vote booths. https://apnews.com/general-news-0951be4c1669492eb19babf38e3f6b3c

[115] Krishna PN Puttaswamy, Christopher Kruegel, and Ben Y Zhao. 2011. Silverline: Toward Data Confidentiality in Storage-Intensive Cloud Applications. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*. 1–13.

[116] 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* 8, 9 (Sept. 2020), 1618. https://doi.org/10.3390/math8091618

[117] Matt Qvortrup. 2005. First past the Postman: Voting by Mail in Comparative Perspective. *The Political Quarterly* 76, 3 (2005), 414–419. https://doi.org/10.1111/j.1467-923X.2005.00700.x

[118] Rahul Rathi. 2019. Effect of Cambridge Analytica's Facebook ads on the 2016 US Presidential Election. https://towardsdatascience.com/effect-of-cambridge-analyticas-facebook-ads-on-the-2016-us-presidential-election-dacb5462155d

[119] Robert W. [R-OH-18 Rep. Ney. 2002. H.R.3295 - 107th Congress (2001-2002): Help America Vote Act of 2002. https://www.congress.gov/bill/107th-congress/house-bill/3295

[120] Manoel Horta Ribeiro, Raphael Ottoni, Robert West, Virgílio A. F. Almeida, and Wagner Meira. 2020. Auditing radicalization pathways on YouTube. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 131–141. https://doi.org/10.1145/3351095.3372879

[121] Gary Robertson. 2022. Plea agreements reached by 4 in NC Congress ballot probe. https://apnews.com/article/2022-midterm-elections-health-covid-north-carolina-raleigh-07c0fbdcdbb4595e4ea0076892c9d294

[122] Bertrall L. Ross and Douglas M. Spencer. 2019. Passive Voter Suppression: Campaign Mobilization and the Effective Disfranchisement of the Poor. *Northwestern University Law Review* 114, 3 (November 2019), 633–704.

[123] Peter B. Rønne, Arash Atashpendar, Kristian Gjøsteen, and Peter Y. A. Ryan. 2020. Coercion-Resistant Voting in Linear Time via Fully Homomorphic Encryption: Towards a Quantum-Safe Scheme. Vol. 11599. 289–298. https://doi.org/10.1007/978-3-030-43725-1_20 arXiv:1901.02560 [quant-ph].

[124] Jeff Sauro. 2011. *A Practical Guide to the System Usability Scale.* Measuring Usability LLC. https://measuringu.com/product/suspack/

[125] Sinem Sav, Apostolos Pyrgelis, Juan Ramón Troncoso-Pastoriza, David Froelicher, Jean-Philippe Bossuat, Joao Sa Sousa, and Jean-Pierre Hubaux. 2021. POSEIDON: Privacy-Preserving Federated Neural Network Learning. In *Proceedings 2021 Network and Distributed System Security Symposium*. Internet Society, Virtual. https://doi.org/10.14722/ndss.2021.24119

[126] Stuart Schechter, Serge Egelman, and Robert W. Reeder. 2009. It's not what you know, but who you know: a social approach to last-resort authentication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1983–1992. https://doi.org/10.1145/1518701.1519003

[127] Uwe Serdult, Micha Germann, Fernando Mendez, Alicia Portenier, and Christoph Wellig. 2015. Fifteen years of internet voting in Switzerland [History, Governance and Use]. In *2015 Second International Conference on eDemocracy & eGovernment (ICEDEG)*. IEEE, Quito, Ecuador, 126–132. https://doi.org/10.1109/ICEDEG.2015.7114482

[128] Adi Shamir. 1979. How to Share a Secret. *Commun. ACM* 22, 11 (1979), 612–613.

[129] Tanusree Sharma, Yujin Potter, Kornrapat Pongmala, Henry Wang, Andrew Miller, Dawn Song, and Yang Wang. 2024. Unpacking How Decentralized Autonomous Organizations (DAOs) Work in Practice. In *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 416–424. https://doi.org/10.1109/ICBC59979.2024.10634404

[130] Divya Siddarth, Sergey Ivliev, Santiago Siri, and Paula Berman. 2020. Who Watches the Watchmen? A Review of Subjective Approaches for Sybil-Resistance in Proof of Personhood Protocols. *Frontiers in Blockchain* 3 (Nov. 2020). https://doi.org/10.3389/fbloc.2020.590171

[131] Smartmatic. 2024. Estonia: The World's Longest Standing, Most Advanced Internet Voting Solution. https://www.smartmatic.com/featured-case-studies/estonia-the-worlds-longest-standing-most-advanced-internet-voting-solution/. Accessed: 2024-12-07.

[132] Warren D Smith. 2005. New cryptographic election protocol with best-known theoretical properties. (Sept. 2005), 14.

[133] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J. Alex Halderman. 2014. Security Analysis of the Estonian Internet Voting System. In *ACM SIGSAC Conference on Computer and Communications Security*. https://doi.org/10.1145/2660267.2660315

[134] Oliver Spycher, Reto Koenig, Rolf Haenni, and Michael Schläpfer. 2011. A New Approach towards Coercion-Resistant Remote E-Voting in Linear Time. In *Proceedings of the 15th International Conference on Financial Cryptography and Data Security*. 182–189. https://doi.org/10.1007/978-3-642-27576-0_15

[135] USA.gov. 2024. Absentee and Early Voting. https://www.usa.gov/absentee-voting. Accessed: 2024-12-07.

[136] Stefan G. Weber, Roberto Araujo, and Johannes Buchmann. 2007. On Coercion-Resistant Electronic Elections with Linear Work. In *The Second Conference on Availability, Reliability and Security*. https://doi.org/10.1109/ARES.2007.108

[137] Wendy Weiser. 2020. How to Protect the 2020 Vote from the Coronavirus | Brennan Center for Justice. https://www.brennancenter.org/our-work/policy-solutions/how-protect-2020-vote-coronavirus

[138] Danielle Wiener-Bronner. 2014. Yes, There Are Elections in North Korea and Here's How They Work. https://www.theatlantic.com/international/archive/2014/03/north-korea-elections-guide/358875/

[139] William Wilder. 2021. *Voter Supression in 2020.* Brennan Center for Justice at New York University School of Law. https://www.brennancenter.org/sites/default/files/2021-08/2021_08_Racial_Voter_Suppression_2020.pdf

[140] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. 2018. *Blockchain Technology Overview.* Technical Report NIST Internal or Interagency Report (NISTIR) 8202. National Institute of Standards and Technology. https://csrc.nist.gov/pubs/ir/8202/final

[141] Margarita Zavadskaya. 2024. Why Does the Kremlin Bother Holding Sham Elections? https://www.journalofdemocracy.org/elections/why-does-the-kremlin-bother-holding-sham-elections/

Appendices – Supplemental Materials

SOSP'25 submission.

Contents

A Broader computing systems applications

Ever since early visionaries such as J.C.R. Licklider [93], generations of technology researchers and engineers have predicted the positive transformative effects that networked, distributed or decentralized computer systems would have on the ways people interact socially, from small groups to entire nations. The USENET was the first large-scale decentralized platform that its users called, and widely believed, would be strongly "democratizing" – at least by virtue of given everyone with Internet access an "equal" and "uncensorable" platform for speech, communication, and coordination with others for myriad purposes. Decades later, after USENET had effectively succumbed to a heat death from uncontrolled spam and been replaced by commercialized platforms, excitement in social media applications in particular sparked another global wave of optimism about large-scale computing technologies being "democratizing", culminating in the short-lived Arab Spring movement [82].

Not long after this movement's collapse, however, it became increasingly clear that large-scale computing platforms were just as readily usable as tools of surveillance and control by anti-democratic actors, as for support or promotion of democracy or related ideals. Increasing suspicion has fallen on large-scale platforms for their potential to interfere with democracy in nontransparent and unaccountable ways, whether intentionally – such as by potentially affecting election outcomes through search-engine rankings [54] – or promoting sensationalistic and radicalizing content in the blind algorithmic pursuit of human attention and associated advertising revenue [120] – or generally increasing the political polarization of a nation by presenting users with information diets consisting mostly of opinions of their friends and others they already agree with [61]. A large segment of the public even in advanced democracies has increasingly been drawn into a "technology backlash" against large-scale computing platforms and the companies and personalities associated with them [80]. In summary, while the early proponents of large-scale distributed computing technologies tended to see only positive effects on society, hindsight now paints a much more pessimistic picture.

Even focusing more narrowly on technologies specifically intended to support and improve democracy, such as e-voting, boundless optimism has generally turned toward dark pessimism. Technology-accepting politicians and the public at large regularly express interest in and even demand the convenience of voting on their favorite mobile devices, while experts in voting practices and security consistently and nearly-unanimously agree that using e-voting is extremely risky and not generally recommended at least at high-stakes, national scales. Worse, large-scale decentralized computing platforms such as Decentralized Autonomous Organizations (DAOs) might well be used not as tools of democracy but rather tools *against* democracy, such as to buy votes at large scale while exposing the attacker to little or no risk of accountability or deterrence [21].

Despite this pessimistic climate surrounding computing technology's interaction with democracy, there nevertheless remains persistent and recurring interest in the idea of *democratic computing systems* in various forms: large-scale distributed or decentralized systems designed to serve and be accountable to their *human* user population, and to support social and democratic coordination and self-organization in the "one person, one vote" egalitarian characteristic of democracy. In the recently-burgeoning interest in DAOs, for example, there has been a significant subcurrent of interest in *democratic DAOs*, in which decentralized organizations operating entirely "on the blockchain" are nevertheless controlled, at least in part, by deliberative decision-making and voting among their human users. These aspirations always run into the fundamental problem of the Sybil attack, however [52] – namely that our networked computing systems do not reliably or security "know" *what a real human is.*

This problem in turn led to a subcurrent of interest in *proof of personhood* protocols and mechanisms – attempting to distinguish real humans from Sybils or bots, in various fashions [63, 130]. Recent lessons in this space have revealed the ultimate importance of not only distinguishing humans from bots, but also of ensuring that participating humans are representing *their own interests* and not those of a coercer, astroturfer, or other vote-buyer, as was found to be gradually taking over the Idena proof-of-personhood system over several years [108]. Thus, when a proof of personhood is intended to facilitate democractic self-organization in any fashion, the lack of any form of coercion resistance results in a gaping vulnerability to vote-buying or astroturfing, whether at small or large scale.

Beyond merely supporting e-voting in conventional elections, therefore, we see the work represented by this paper as a potential stepping stone or building block for true large-scle democratic computing systems in the future: distributed or decentralized systems governed by, and accountable to, their users collectively, while assuring that their voting participants are actually representing their own interests and not someone else's. While the details of such applications of this work to democratic computing we of course must leave to future work, one conceivably-workable approach is to combine TRIP's in-person coercion-resistance mechanism with the in-person practice of *pseudonym parties* as a proof-of-personhood mechanism, where people obtain one anonymous participation token each at periodic real-world events [34, 64].

If this key socio-technical hurdle – of identifying *real people acting in their own interests* – can be surmounted, then there is already ample evidence that further development could enable distributed computing systems to improve and enhance democracy in numerous other ways long envisioned but as-yet of limited practicality. Even serious political philosophers and democratic theorists see this potential, as exemplified in Hélène Landemore's proposal of "Open Democracy" for example [74]. Similarly, for over two decades there has been a persistent subcurrent of interest in concepts such as *liquid democracy* – enabling large numbers of users to participate more regularly and directly in governmental deliberation and voting as their time and interest permits, by participating directly in selected topics or forums close to their interest while *delegating* their vote in other areas to others they trust to represent them, individually rather than only in large collectives as in traditional representative democracy [32, 62]. This idea took hold for a number of years in the German pirate party, which implemented and used this idea for intra-party deliberation via their LiquidFeedback platform [84] – although they never adequately solved the conundrum of wishing to support user anonymity and freedom of choice while ensuring that only *real humans* were voting, one per person, and only in their own interests. Similarly, widespread and increasing interests in *mass online deliberation* and scalable participatory "mini-publics" [74] shows widely-acknowledged promise to help make democracy more effective and participatory, but similarly requires as a key building block a mechanism to ensure that only real humans are registered and participating in their own interests, free of coercion.

Even the recent furor of global interest and research in machine learning, artificial intelligence, and large language models in particular, often neglects to observe that these technologies build on – and are fundamentally dependent upon – the *collective intelligence* (CI) of all the human users who, intentionally or not, created and provided the enormous global datasets of data that the large AI models were trained on. Further, the inability to distinguish human-generated from artificially-generated content presents an increasing threat not only to artists and other content produces [18, 40] but also to the long-term effectiveness and usability of AI systems such as LLMs themselves [17]. Without being able to tell what comes from people and what comes from bots, Sybils, or AIs, we may lose not only the online world's effective anchor on "truth" or "reality" but also the effectiveness of the very AI systems of such great interest at the moment.

In summary, while we fully believe that coercion-resistant e-voting along the lines this paper explores is a useful, challenging, and interesting systems research topic in its own right, we also see it as ultimately far more relevant in the long term as a potential enabler for future democratic computing systems and human-based collective intelligence platforms, which might conceivably serve and be used by collectives of all sizes from small groups to entire nations or ultimately even the global population of networked computing users.

B The baseline standard of in-person voting

The only approach to national voting in democracies today enjoying near-universal acceptance by experts is in-person voting with paper ballots. Marking paper ballots in a privacy booth at a controlled polling site helps protect ballot secrecy against coercion or vote buying. This protection is imperfect, as it can be compromised for example by voters taking "ballot selfies" or recording themselves in the ballot booth [26], as further discussed later in Appendix E. Even with such weaknesses, the promixity of officials remains at least a psychological deterrent against on-site attempts at coercion or voting under improper influence. Supporting transparency, voters normally deposit their marked ballots into an on-site ballot box themselves, and independent election observers build "end-to-end" transparency by monitoring the subsequent handling and counting of the ballots.

Electronic *ballot marking devices* or BMDs in voting booths can assist voters with special needs such as screen reading, and can print an error-free paper ballot only after validating the voter's choices, addressing usability issues of hand-marked paper ballots [31, 37]. Voters still normally use BMDs alone in a privacy booth, however, to preserve ballot secrecy. BMDs maintain transparency by printing a paper ballot that the voter can (and arguably is *expected* to) inspect and verify for correctness before casting. These printed ballots might be counted mechanically, but they leave a tamper-evident "paper trail" subject to independent observation and ballot-handling practices such as manual recounts or risk-limiting audits [30]. BMDs leave unchanged the usual requirement that voters vote *in-person*, however, only at official polling sites.

The inconvenience and sometimes impracticality of in-person voting creates pressure to adopt remote processes, such as postal voting, which allow voters to vote from anywhere. Because the election authority cannot control arbitrary locations, however, remote voting normally compromises ballot secrecy and hence resistance to coercion or vote-buying. This weakness is not purely theoretical: for example, absentee ballot fraud in North Carolina was found to have altered election outcomes [9, 70]. Despite this weakness, convenience and voter-turnout considerations have led a few countries, such as Switzerland, to adopt remote processes such as postal voting as the normal-case approach [95].

C Registration and Voting Channels

For clarity of design and exposition, the main paper simplistically assumes that there is a single *registration* step that is mandatory for all voters, and that e-voting is the only voting *channel*, or supported method of ballot casting.

Deploying Votegral or any e-voting architecture in a real nation would of course require addressing many important considerations to integrate and adapt e-voting to that nation's legal, cultural, and practical environment. These considerations would necessarily include adapting Votegral to the nation's prevailing voter registration paradigm, and integrating Votegral properly with the nation's pre-existing voting channels. Addressing such challenges in detail for any given nation is beyond the scope of this paper, but we briefly outline some of the relevant considerations here, with a few particular nations as illustrative and contrasting examples.

C.1 Voting across multiple channels

In practice, e-voting is never realistically deployed (at least nationally) as the *sole* available voting channel. Instead, e-voting is generally an opt-in alternative to one or more pre-existing "base case" channels, such as in-person voting with paper ballots and/or postal voting. We outline these voting channel considerations by briefly examining practices in three countries: the US, Estonia, and Switzerland.

US: Although in-person voting by paper ballot remains the established baseline voting channel throughout most of the US, all states allow certain voters to request and submit *absentee* ballots by post, under conditions varying by state [5, 111, 135] Some states allow some or all voters to join a *permanent absentee voting list* and obtain mail-in ballots automatically before every election [101]. Eight US states have adopted postal voting as a primary voting channel, with other states allowing postal voting for all voters but only in certain districts or for certain elections [100]. Although the US has never systematically adopted any remote e-voting system, many states allow certain citizens, such as military and those with disabilities, to submit scanned ballots by e-mail [5]. As a communication channel that is neither encrypted nor even authenticated by default, this *ad hoc* adoption of e-mail as the US's standard "remote e-voting system" (while carefully never calling it one) seems like a strikingly worst-of-all-worlds solution to the demand for e-voting.

Estonia: Estonia introduced e-voting as an opt-in alternative to in-person voting with paper ballots in 2005, and its adoption has gradually increased from 2% to over 50% [131]. Despite e-voting's popularity in Estonia, casting a paper ballot remains an option. Further, these two alternative voting channels contribute further to Estonia's revoting-based approach to coercion resistance. A paper ballot cast in-person overrides any e-vote cast remotely by the same person, and the in-person voting deadline is later than the e-voting deadline. A voter who is coerced (or has their e-voting materials confiscated) all the way though the e-voting deadline, therefore, still has the option in principle of casting an uncoerced ballot in person. The effectiveness of this "last line of defense" against coercion remains limited, however, by the voter's practical ability to make an in-person trip to a polling station, in secret without the knowledge of the coercer, during the short time window between the e-voting and in-person voting deadlines. For voters living outside Estonia or living full-time with abusive partners, for example, it seems unlikely that the in-person backup channel can offer an effective defense.

Switzerland: Although Switzerland has so far adopted e-voting only in a few cantons, this adoption came amidst a backdrop of *two* well-established voting channels already available almost nationwide: namely both in-person voting and postal voting [95]. Since postal voting was already the most popular baseline channel in Switzerland before the adoption of e-voting, the general security and privacy properties of this postal baseline effectively came to define the "goal posts" for Switzerland's e-voting program. This historical development explains in part why coercion resistance has never been a high priority in Switzerland's e-voting program: the public and political circles alike had already adopted and accepted postal voting almost universally, despite its well-known lack of coercion resistance. Thus, an e-voting system design without coercion resistance was at least no *less* coercion-resistant than the predominant baseline of postal voting.

C.2 Registering *to vote* versus *for e-voting*

The e-voting research literature, and especially that addressing coercion resistance using fake credentials [41, 78], typically assumes that all voters must perform a step generically called "registration". In realistic environments supporting multiple voting channels discussed above, however, the reality of what "registration" might mean is more complex.

In particular, we must distinguish in practice between two potential forms of "registration": namely *registering to vote*, and *registering for e-voting*. "Registering to vote" means signing up to be allowed to cast a ballot in an election *at all*, by any voting channel. "Registering for e-voting" means signing up to use the e-voting channel in particular, as distinguished from any other available voting channels.

The Votegral architecture makes an important and essential assumption that voters are required to register in-person at least *for e-voting*, in particular acquiring real and fake credentials for coercion resistance. Votegral makes no significant assumptions about whether or how voters register *to vote*, however. While we generically use the term "registration" in the main paper for consistency with the e-voting literature, perhaps a better terms for the in-person step that Votegral requires might be "e-voting sign-up" or "credentialing" rather than "registration." The actual requirements for and considerations surrounding Votegral's "registration" stage are more appropriately comparable to that of obtaining or renewing an identity document at a government office, and less suitably comparable to "registering to vote" in the US for example. We next briefly examine these contrasting notions of "registering to vote" versus "registering for e-voting" (credentialing) in the contrasting contexts of Europe versus the US.

Europe: In most European countries, including the above examples of Estonia and Switzerland, a standard and accepted responsibility of government is to maintain an accurate registry of inhabitants: i.e., a database listing who currently lives where, including basic metadata such as age, origin, and citizenship. Residents are expected and required to report when they move into, out of, or between localities, for various reasons orthogonal to voting, such as for reliable communication between the resident and the government, and for determination of the resident's tax obligations. Determining which current residents of some locality are eligible to vote in which elections is just one more, of many, standard and required uses of this registry of inhabitants. Since governments are expected to "know their inhabitants" anyway, including residence-address and voting-eligibility metadata, "registration to vote" is generally automatic throughout European countries. Unlike in the US, voters need not do anything special to "register to vote": instead they receive instructions and materials needed for voting automatically by virtue of being listed on the appropriate registry of inhabitants.

Votegral's assumption of in-person "registration" might therefore seem at first glance incompatible with Europe's convenient standard of automatic registration to vote. This is not the case, however, once we correctly identify Votegral's "registration" step as *registration for e-voting* or *credentialing*. If Votegral were to be integrated into a European nation's election system, in particular, it need not and should not affect the European standard of automatic registration to vote. Instead, in-person credentialing would be a step required only for those voters wishing to opt-in to the *e-voting channel* in particular, and need not be required of anyone opting for existing voting channels such as in-person voting by paper ballot. Thus, in-person registration for e-voting or credentialing would remain an occasional convenience cost, but only one imposed on those desiring the option of using the e-voting channel. Visiting a government office in-person is often still required occasionally, anyway, for other critical processes such as obtaining or renewing IDs or passports. European voters could therefore in principle do their in-person credentialing for e-voting at the same time as they sign up for or periodically renew other IDs.

US: In the US, in contrast, voters must normally register to vote as a separate step, as a prerequisite to casting a ballot via *any* voting channel. In states where this mandatory voter registration step is normally done (or required to be done) in-person anyway, there is a conceivable opportunity to deploy an e-voting architecture like Votegral without imposing on voters any "new" obligation to visit a government office in most cases. US voters in principle could opt-in to Votegral and perform the necessary credentialing for e-voting immediately after registering to vote, in the same in-person visit.

Despite this potential short-term opportunity, however, we do not suggest or endorse such an approach as an ideal long-term strategy. The US's requirement of a separate voter registration step is not only inconvenient to voters, but has been regularly abused in practice as a method of deliberate voter suppression [73, 122]. The fact that the roster of registered voters (often including party affiliation) is publicly available in many states further exacerbates this weakness. In particular, any coercer wishing to ensure that certain "undesirable voters" do not vote at all need not monitor the target voters continuously: the coercer need only watch the public lists of registered voters and threaten "consequences" if any of the targets' names appear on it. Thus, independent of the coercion-resistance properties of any particular voting *channels*, the US's voter *registration* paradigm has severe inherent weaknesses to coercion *not to register at all*, at the very least.

The hypothetical adoption of an e-voting architecture like Votegral in the US would not appear to affect these US-specific registration issues either for better or worse in any obvious way. Votegral would not help address the problem of voter suppression at registration time, but it would not appear to exacerbate the problem either. Further, realistic solutions to the US's registration-time voter suppression weaknesses appear to be orthogonal to Votegral and in general to any particular voting channel. The obvious solution is to adopt the European approach of automatic voter registration: there is no opportunity for coercers to suppress voters at registration time if all eligible voters are registered automatically without their taking any specific action. This solution clearly requires no new technology such as e-voting, and hence we argue is important but orthogonal to and beyond the scope of this paper.

C.3 Cross-channel coercion considerations

Any election system supporting multiple channels and desiring coercion resistance must in general consider coercion-resistance properties not just within but across channels. We noted about in Appendix C.1 how Estonia's design consciously uses in-person voting as a backup channel to e-voting, including for purposes of coercion resistance.

In general, a voting system may be effectively only as coercion-resistant as its *least-coercion-resistant* voting channel, unless the system's cross-channel interactions are carefully designed to avoid this undesirable effect. As a purely-hypothetical negative example, suppose that Switzerland were to adopt a coercion-resistant e-voting channel such as Votegral, without addressing the legacy problem that Switzerland's currently-predominant postal-voting channel is not coercion-resistant. Even if both the e-voting and in-person voting channels are perfectly coercion resistant, a coercer could simply demand that a target voter elect postal voting, and supervise the filling and mailing of the victim's postal ballots. In other words, a voter's *choice of voting channel* itself represents a point of potential vulnerability to coercion.

One way to address this cross-channel issue in general is to allow votes via less-coercion-resistant channels to be overridden by votes via more-coercion-resistant channels. Estonia's cross-channel design is an example of this design principle, albeit with important time-based constraints due to the revoting approach. An important challenge with revoting and vote-overriding in general, however, is ensuring the voting system's end-to-end transparency, not just in terms of the verifiability each channel provides, but also verifiability of correct tallying *across* channels. If Alice revotes several times using Estonia's e-voting channel, then overrides all of those votes by casting a paper ballot in-person, how do election observers—or the public at large—know that only Alice's in-person vote was counted? If the government makes publicly-available the fact that Alice cast a ballot in-person, then this defeats coercion resistance because Alice's coercer can see that information as well. If the government keeps this information secret, in contrast (as Estonia's must), then the public must apparently "just trust" that Alice's last e-vote and her in-person vote are not *both* counted, no matter how transparent the e-voting channel alone might be. (Estonia's current e-voting channel itself is not verifiable anyway [133], so while this cross-channel transparency issue is important in general, it is moot in Estonia's current practical design.)

In summary, a nation's registration paradigm and other voting channels represent are important environmental issues in any conceivable deployment of a coercion-resistant e-voting system like Votegral. Both the details of the environment and the appropriate solutions are necessarily country-specific, however, and hence are necessarily beyond the scope of this paper.

D Credential Lifetime and Surprise Coercion

Any deployment of a system like Votegral faces a policy choice of how long e-voting credentials should remain valid before expiring and requiring renewal. While making this policy choice in a given environment is beyond the scope of this paper, we briefly examine here a few credential-lifetime considerations in general, and their important interactions with coercion resistance in particular.

D.1 Potential principles for credential lifetimes

Usability considerations alone might suggest allowing credentials to be used indefinitely without ever expiring. Many of the multitude of reasons that identity documents such as passports usually have expiration dates, however, apply similarly to the e-voting credentials that TRIP produces. Like any important security token or identity proxy, TRIP credentials might fall into the wrong hands for numerous reasons, their security might erode due to weaknesses gradually discovered in the cryptographic algorithms they depend on, etc. Imposing an expiration date on e-voting credentials is simply good security practice, for standard reasons not specific to TRIP.

A more arguably-reasonable choice of credential lifetime, therefore, might be to coincide with the lifetimes of other government-issued identity documents that a citizen may need to renew occasionally in person, such as passports, driver's licenses, or other ID cards. Such an alignment of lifetimes across identity documents could in principle save voters' time and reduce convenience costs by allowing voters to renew multiple documents at once during the same government office visit, in the common case. This benefit makes the important presumption, of course, that the government is sufficiently well-organized to ensure that a single government office can offer these distinct renewal services – or at least, multiple offices geographically colocated at the same site.

D.2 Surprise coersion threats

An important issue with long credential lifetimes, however, is the threat of *surprise coercion*. An average voter may arguably be unlikely to understand or appreciate the risk of coercion sufficiently to have taken adequate precautions against this threat, before being "surprised" by a coercer who suddenly appears and demands immediate compliance under some threat (or offering financial rewards for compliance).

An obvious surprise coercion scenario in the case of TRIP is that a voter printed only one real credential and no fake credentials at the time of last registration, and thus has no fake credentials prepared to give or sell to the coercer. If the coercer guesses, perhaps accurately, that most voters are likely in precisely this boat – at least before they have ever actually faced a coercion threat – then the coercer might simply demand their real credential or whatever voting device it is installed on, and take it as "likely enough" that this is indeed the voter's real credential. After surprising the voter this way, the coercer might successfully forbid the voter from re-registering at least until the voter's current credentials expire, since we consider it infeasible to hide registration events from coercers for reasons detailed in Appendix F.

Long credential lifetimes thus risk increasing the amount of time a successful surprise-coercion attack succeeds in allowing the coercer to control or vote on behalf of the victim, before a government-imposed credential-renewal obligation forces the coercer either to permit (or require) the voter to re-register in order to continue e-voting at all. Shorter credential lifetimes are thus preferable from a coercion resistance perspective, in order to reduce this window of vulnerability that many ordinary voters might have to surprise coercion.

A long-term coercer could of course forbid a victim from *ever* registering even to renew expired e-voting credentials. In a realistic environment with multiple voting channels properly designed to ensure coercion resistance jointly and not just individually, as detailed in Appendix C, forbidding the victim from ever renewing e-voting credentials should effectively leave a fallback path to a backup channel such as traditional in-person voting – at least if any backup voting channel is effectively available to the victim. In a worst-case scenario where a coercer can successfully forbid a victim from *ever* visiting a government office, whether to register for e-voting or to vote in person, we consider this tantamount to long-term imprisonment of the victim. This unfortunately may occur in extreme circumstances, but probably no realistic approach to coercion resistance can address it.

D.3 Random or deniable renewal obligations

Although we cannot realistically hide registration events as discussed in Appendix F, a potential defense against surprise coercion might be to make credential lifetimes *unpredictable*. In addition to imposing a fixed upper bound on credential lifetime, for example, the government might periodically select a subset of voters by random lottery, notifying them that their e-voting credentials will expire within a few months and must be renewed for continued use of e-voting. Even a voter with brand-new credentials would be subject to this lottery.

While such a proactive credential-expiry lottery would of course inconvenience the unlucky selected voters, it would also have the global benefit of providing "cover" for the government to force early renewal of e-voting credentials for other reasons as well: such as due to an anonymous tip-off from a coercion victim or the victim's friend that the victim needs a plausibly-deniable, ostensibly *government-initiated* reason to visit the registration office, re-register, and perhaps report the coercion incident in relative safety. Such an "anonymous tip-off" mechanism might in principle even be built into client-side voting software. Such a mechanism would be analogous to the way that some banks issue customers with *duress PINs* that they can use to inform the bank silently if they are coerced to withdraw money at an ATM [72].

D.4 On-demand fakery versus preparation

Some prior coercion-resistant e-voting schemes such as Civitas [41] have the attractive property that a voter can, at least in theory, cryptographically create new fake credentials at any time. This *on-demand* creation of fake credentials is sometimes cited as a reason to prefer such designs over a design like TRIP, which requires voters to "prepare in advance" by obtaining fake credentials at registration time before coercion occurs, and otherwise leaves voters vulnerable to surprise coercion as discussed above.

Closer analysis of realistic suprise-coercion scenarios in the context of realistic, arguably-usable deployments of such systems, however, reveal this apparent advantage of Civitas-like designs to be more theoretical than practical. The only practical embodiments of Civitas-like designs ever subject to a user study [103], however, rely on trusted hardware to keep the voter's real credential safe and issue fake credentials on demand, e.g., on entry of incorrect PINs to the trusted smart card [55]. In such a deployment context, we may consider it a near certainty that an unprepared voter subject to surprise coercion will have obtained only one such trusted smart card, which the coercer can then simply demand and confiscate.

Without the assistance of *some* electronic device that holds the voter's real Civitas credential, and which we must assume the voter has somehow hidden and protected from the coercer despite being entirely unprepared for coercion, the theoretical possibility of on-demand creation of Civitas credentials essentially relies on the voter himself being a cryptographer. This hypothetical voter must in fact be not just an average cryptographer, but a genius in the league of the legendary Indian mathematician Ramanujan, who could perform complex mathematical calculations on large numbers in his head without the aid of any electronic device. Needless to say, we find it implausible that this "Ramanujan voter" profile fits any significant percentage of real voters anywhere on Earth.

Because the trusted hardware assumed by usable Civitas-like schemes is undoubtably much more expensive than the paper credentials that TRIP uses, it seems likely that typical voters unprepared for coercion will be much more likely in practice to have only one smart card in a Civitas-like design than a typical TRIP user is to have obtained only one real paper credential. TRIP's user-facing design, in fact, uses the term *test credentials* in place of *fake credentials*, and informs voters that test credentials may be legitimately used for purposes other than coercion – such as for casual testing of the voting system and for educational purposes, such as to show children or friends not yet eligible to vote how the e-voting system works. By making fake/test credentials not only inexpensive but also in principle worth the trouble of creating for reasons entirely orthogonal to worries about coercion, TRIP's design attempts to increase the chance that many or even most voters will create one or more fake credentials when registering, just "for fun" if for no other reason.

E Devices and Anti-Recording Policies

Even in accepted in-person voting processes using paper ballots, personal devices capable of recording can already compromise the coercion resistance that privacy booths are intended to provide at official polling sites. The use of recording devices can similarly compromise the effective coercion resistance of Votegral's credentialing process in the registration booth, in essentially the same fashion. Just as a coercer can already demand that a voter photograph themselves with their ballot marked according to the coercer's preferences in a voting booth today, a coercer could similarly demand that a voter record themselves creating their real and fake credentials in TRIP's privacy booth, showing the coercer in particular how the voter marked their real credential.

The fundamental threat that recording devices impose on registration in TRIP thus appear largely equivalent to the corresponding threat of recording devices used in privacy booths during in-person voting: it seems that Votegral neither solves this problem nor makes it worse. Nevertheless, it is worth briefly exploring the background and current prevailing situation surrounding this challenge for in-person voting, then outlining potential policy and enforcement options that might be applicable to e-voting registration using TRIP.

E.1 Ballot selfies and recording prohibitions for in-person voting

A common practice is to take a *ballot selfie* in the voting booth showing the voter's marked ballot, then perhaps posting it to social media. This practice, while usually intended innocently to demonstrate civic engagement, could easily double as a "proof" of how one voted to satisfy a coercer's demands or to participate in a vote-buying scheme. As a result, ballot selfies and related photography of marked ballots is at least legally prohibited in some countries as well as a dozen US states [49, 81, 114].

Even where ballot selfies are legally prohibited, the level enforcement and potentially-applicable penalties vary. Many jurisdictions do not enforce their ballot selfie ban at all unless there is evidence of coercion, vote buying, or other fraudulent activity. In most US states that ban ballot selfies, being caught taking one is a misdemeanor punishable by a fine, although in Illinois it is a felony punishable by 1–3 years in prison [4].

Even where ballot selfies are illegal and even strongly punishable, that does not of course mean that voters cannot or will not secretly take them anyway – if an abusive spouse, local gang member, or other strong coercer threatens violence or other significant repercussions if the voter does not, for example. The deterrent represented by *potential* punishment by the government, only if participation in coercion or vote buying is actually caught, does not necessarily outweigh *likely* or *certain* punishment by a strong coercer whose demands go unfulfilled. Thus, today's accepted coercion-resistance practices appear to go only so far and are clearly weak against stronger forms of coercion, and unfortunately we know of no systematic evidence of how prevalent such strong coercion might be. We hope it is extremely rare, but it is also possible that it is more common but just rarely *reported*—because in such cases the coercer usually "wins" and successfully intimidates the voter not only into following the coercer's demands but also into never revealing the incident.

In principle, in-person polling sites could address stronger coercion *proactively* by requiring all voters to check personal devices into lockers before entering the voting area containing the privacy booths. Polling sites could even enforce such a recording device ban by requiring voters to enter the voting area through airport-style metal detectors after checking all personal devices, just as visitors often must already do at many high-security government sites such as embassies and military bases. Such strong measures would incur high costs, however: financially, logistically, in terms of convenience and perceived friendliness to voters, and potentially in terms of negative impact on voter turnout. The cost to all voters and to the election authority thus seems unlikely to be justified by the incremental protection such measures might provide against apparently-rare incidents of actual coercion or vote buying. As a result, we are not aware of any government that has actually imposed such extreme measures at ordinary polling sites in the normal case for in-person voting.

E.2 Anti-recording considerations for TRIP

A government considering the adoption of a coercion-resistant e-voting system like Votegral would need to decide whether to impose a ban on the use of recording devices during a TRIP registration session in a privacy booth, and if so, to what degree to enforce such a ban. We consider this policy decision to be important but largely orthogonal to the technical focus of this paper. Many of the considerations and tradeoffs outlined above for in-person voting seem likely to translate directly to corresponding tradeoffs for TRIP privacy booths.

It is relatively inexpensive and practical simply to forbid the use of recording devices in TRIP privacy booths, to inform voters of this ban in educational materials before they enter the booth, and to impose penalties as a deterrent to noncompliance with the recording ban. As with ballot selfie bans for in-person voting, such prohibitions may be effective in deterring most casual violations, but may not be effective against stronger forms of coercion such as the abusive partner or gang member demanding a secret recording "or else."

A government concerned with strong coercion could of course take stronger proactive measures, such as requiring voters to check personal devices and pass through a metal detector before entering a privacy booth for TRIP credentialing. For the same cost and related reasons such measures are already rarely if ever taken for in-person voting, however, we do not expect such measures to be common in a practical deployment of Votegral. Regardless, this important consideration remains a policy decision rather than a technical one.

We envision there may be situations in which stronger anti-recording measures may be cost-effective, however. One prominent use-case for remote e-voting is for expatriates or military members living overseas. Expatriates often need to visit an embassy or consulate of their nationality at least once every few years anyway to renew their passport, and embassies are typically considered high-security establishments that often require visitors to check personal devices and enter through metal detectors anyway. If a government were to deploy Votegral so as to make TRIP privacy booths and kiosks available to expatriates at embassies and consulates, then there may be little if any extra cost to imposing stronger anti-recording measures at these sites. Similarly, military members are often stationed at, or in general the vicinity of, a military base that has strongly-protected areas for other othogonal reasons anyway, which could likewise double as secure sites for TRIP registration at low incremental cost.

E.3 Assumed limitations of ordinary voters

Our assumption above that the use of personal devices is at least forbidden in, if not proactively excluded from, registration booths, affects TRIP's design for coercion resistance in other more subtle ways.

In particular, TRIP's design assumes that ordinary voters *without the help of an unauthorized personal device* cannot readily perform cryptographic calculations in their heads. (If a coerced voter *does* have and is successfully coerced to use an unauthorized device, we expect a recording attack as discussed above is the simplest and most likely threat vector.)

A rare genius to whom this assumption about the limitations of ordinary voters does *not* apply, we call a *Ramanujan voter*, in reference to the legendary Indian mathematician Srinivasa Ramanujan, who may be one of the few historical examples of a real person potentially capable of such feats.

If a victim of coercion is such a hypothetical Ramanujan voter, then the coercer could readily defeat TRIP's coercion resistance by demanding that the voter mentally decode the QR code containing the real credential's IZKP commit once it is printed, then compute a cryptographic hash of that commit, and use the resulting hash in selecting an envelope from the stack of those available in the privacy booth: e.g., such that the cryptographic challenge matches a certain number of prefix bits from the envelope's cryptographic hash. The coercer could still not be certain, but could obtain some probabilistic confidence, that the revealed "real credential" is indeed real, because when printing fake credentials the Ramanujan voter cannot see the kiosk's commit (in order to hash it mentally) before having to choose an envelope.

This theoretical attack has inspired related prior proposals, using IZKPs for in-person voting, to conceal the real credential's printed commit until the voter's challenge is chosen [99]. While TRIP kiosks could certainly be designed with such mechanical protections, we do not consider such a defense to be necessary or cost-justified in practice. Ramanujan voters capable of performing such feats seem likely to be vanishingly rare in practice – especially considering that this attack only conceivably matters to voters *actually under coercion*, a hopefully-rare event in the first place.

F Impersonation Attacks and Defenses

Any government process that involves identity checking, such as TRIP's in-person e-voting registration process, is potentially vulnerable to *impersonation* attacks. Anyone able to impersonate a voter in registering for e-voting can potentially steal the target voter's real credential and subsequently cast real votes in place the victim.

While we do not expect it is feasible to prevent impersonation attacks entirely in most practical situations, our pragmatic goal is to ensure that successful impersonation is promptly *detectable* by the genuine voter, who can then report and rectify the successful impersonation by re-registering in person. Ensuring that impersonation is detectable implies that TRIP must follow common security practices by *notifying* voters of registration events. This notification requirement in turn implies that we cannot realistically keep the *act of registration* secret from potential coercers, as we might wish to in order to maximize coercion resistance. We discuss these issues and the resulting tradeoffs in the rest of this section.

F.1 Some impersonation scenarios of concern

One situation in which it is hard to prevent impersonation attacks is when an impersonator happens to look "enough" like the victim to pass as the victim when the registration official checks the voter's identity during TRIP check-in. Identical twins are ideally positioned for such impersonation attacks. Many people have and occasionally encounter proximate "look-alikes" to themselves in the real world, however. Such a look-alike who identifies and stalks a target voter sufficiently to learn their name and other basic demographics might be able to trick a registration official's identity check. We do not expect either of these attacks to be common, but we do expect them to occur (and succeed) occasionally.

A variation on the above scenario is if an impersonator does *not* look enough like the victim to pass an identity check, but instead presents a forged identity to the registrar, who does not check the identity closely enough to detect the forgery. We again do not expect this situation to be common in practice, as being caught presenting a false identity in an important government process is strongly punishable almost everywhere, representing a strong deterrent to such uses of fake identity documents, even if they might be fairly likely to go uncaught.

A third scenario we are concerned with, however, is if a particular *registration official* is corrupt, and deliberately accepts faked (or no) identity documents in collusion with an impersonator. The corrupt official might indeed *be* the impersonator. A corrupt registration official may in principle be ideally positioned to impersonate not just one but many voters, in fact, choosing some time other non-colluding officals are absent – e.g., outside the registration office's opening hours – to "register" for e-voting on behalf of any number of voters who the corrupt official believes are unlikely to notice the impersonation. Even if such a corrupt registration official faces severe consequences if caught, the potential scalability of such impersonation attacks, and the potentially-scalable financial rewards (e.g., bribes) for participating in such attacks – or alternatively, potentially-severe punishments for *not* participating under threats of violence by local criminal organizations for example – make such scalable impersonation attacks plausible and even likely in certain environments.

F.2 Detecting impersonation via notifications

Scenarios like those above lead us to conclude that it is unrealistic and unwise to assume that impersonation attacks will *never* occur, or succeed. The common and widely-accepted security "best practice" of *notifying* users of security-critical events, however, is applicable to TRIP as a measure that can help ensure that impersonation attacks, when they occasionally do succeed, are at least promptly detected by the victim. On successful detection of such an impersonation attack, the victim can report the incident and re-register in-person for e-voting, thereby invalidating the impersonator's illegitimate voting credentials. In the process of reporting the incident, the victim also hopefully places (non-corrupt) local registration officials on alert for any future such impersonation attempts.

Since governments typically have at least a mailing address for all eligible registered voters, one obvious default notification channel is by mail. We envision the voter receiving a letter or post card a few days after any successful registration, informing them that the registration occurred, and advising them to contact the registration office immediately in the hopefully-rare event it was not the voter who registered.

An alternative notification channel is digitally, via an E-mail or push notification that the registrar's backend services automatically sends to any voting device(s) the voter has recently registered with or used for voting using any credentials (real or fake) associated with that voter. Ideally, a successful registration for e-voting would notify voters of the registration via both, or all, available notification channels.

A potential weakness with these and other notification channels is if a corrupt registration official can control or suppress this notifcation, or can collude with other insiders capable of doing so. We expect such attacks involving impersonation *coordinated with* notification suppression should be significantly harder and riskier to pull off even than "lone" corrupt official attacks suggested above, since in most cases the official sitting at TRIP's check-in desk to verify voters' identities is unlikely to be the same official (or have the same required expertise) as the officials who control the notification channels (e.g., IT workers). Nevertheless, collusion is not inconceivable, especially in the event of bribery or threats by powerful organized crime in the area, and both postal and digital notification channels tend to be single points of failure/compromise that might conceivably be susceptible to suppression.

F.3 Detection via periodic ledger checking

Provided the election authority's backend services are split across several independent servers run by independent teams, as discussed in §4.1, and provided not *all* of these servers or teams are compromised as Votegral's threat model assumes, there remains at least one further means by which voters might learn of successful impersonation attacks reasonably promptly, even if proactive notification channels such as those above fail or are successfully suppressed.

Each voter's client-side device normally "watches" the ledger (public bulletin board) anyway, periodically checking it for any updates relevant to the voter, whenever the voting application is launched or left running. Thus, if the voter launches the voting application some time after a successful impersonation attack – perhaps to check how they voted in the past as discussed in Appendix M.1, perhaps to vote in an upcoming election, or perhaps for other reasons such as to obtain information or demonstrate the voting system to a family member or friend – this (or any) launch of the voting application represents an opportunity for the client software to "notice" a new registration of the voter appearing on the ledger's registration log, and notify the voter of that successful registration.

If the voting application is run on a device such as a desktop that allows the application to maintain a background "daemon" that occasionally wakes up and proactively checks the ledger (even if only once per day or once per week, for example), then such periodic checks could reduce the duration with which successful impersonation might go undetected, even if proactive notifications are successfully suppressed.

F.4 Coercion resistance implications

All of the above defenses to impersonation attacks, however, underline an important property of Votegral's threat model: we assume that *the act of registering for e-voting* – or re-registering for any purpose, such as to renew credentials or to recover from an impersonation attack – is public information and is not secret. This threat-model assumption unfortunately implies, at a fairly fundamental level, that we cannot realistically hide *the act of registration itself* from potential coercers. This means for example that if a coercer has successfully obtained a victim's real credential, and has forbidden the victim from re-registering, the coercer can see and punish a victim who disobeys this demand. The successfully-coerced victim will have the opportunity to recover from this coercion the next time the victim's credentials expire by government policy and must be renewed – but the victim may *not* have an opportunity to recover until then, if the coercion threat persists for the long term (e.g., in an abusive-partner scenario).

Any attempt to target registration notifications more narrowly – e.g., sending push notifications only to voting devices on which *real* credentials have been activated – risks failing to notify the genuine victim if an impersonator has taken over the victim's real credential, and also risks undermining coercion resistance by creating a new way a coercer might distinguish fake credentials from real. Thus, we do not see any realistic middle ground between *always notifying* and *never notifying* a voter of successful registration events, via any and all available notification channels.

The obvious potential design alternative, then, is *never* to notify voters of registration events, and hence to hide such events from voters and coercers alike. While conceivable, such a design choice appears to have severe negative consequences for transparency in general, and in particular for ensuring that voters have any realistic defense for the impersonation attack scenarios discussed above. In particular, such a secret-registration design places registration events in a fundamentally non-transparent context in which there appears to be no conceivable way for the voter, or anyone else, to notice that an improper registration event has occurred. Voters must individually, and at large, simply trust that registration officials never attempt to register on behalf of voters in order to steal their real credentials and silently manipulate the vote, potentially at scale. While we are obviously concerned with coercion resistance, that being the focus of this paper, we do not believe the threat of coercion is common or important enough, in almost any conceivably realistic election environment, to justify the transparency cost of making it even potentially feasible for a corrupt registration authority to impersonate their own voters at scale without detection.

F.4.1 Coercion to abstain from e-voting

Given the impracticality of hiding e-voting registration events as discussed above, this leaves us with the issue that a long-term coercer can demand that a voter refrain from registering for e-voting at all, and simply watch the public registration log to ensure the voter complies. This perhaps-unavoidable weakness does not mean that a long-term coercer can force the voter to abstain from *voting*, however.

As discussed in Appendix C, in most realistic scenarios, e-voting via Votegral would normally be deployed alongside at least one other "baseline" voting channel such as in-person voting. With proper coercion-resistant cross-channel integration, a long-term coercer who successfully forbids a voter from ever registering for e-voting denies the voter only *the convenience of e-voting*, forcing the voter instead to vote in-person for example. This can still be a problem if the coerced voter cannot readily travel to an in-person voting site any time within the voting period, of course – but this is an existing problem that e-voting at least does not make worse.

In a hypothetical future government wishing to go "all-digital" – as Estonia seems to aspire to – we might envision e-voting at some point being the *only* available voting channel. In such an environment, successful coercion to abstain from registering for e-voting would indeed by default translate into coercion to abstain from voting at all. In such an all-digital environment, however, one potentially-realistic mitigation would be to make e-voting registration mandatory. Citizens would be expected and required to maintain valid e-voting credentials, regardless of whether or how they use them to vote. Periodic mandatory renewal of e-voting credentials might in principle be simply part of an occasional mandatory in-person "citizenship checkup": e.g., an opportunity for the government to verify that the voter is still alive, at the very least, to provide a safe opportunity to report coercion attempts or other issues that might arise, to renew other identity documents at the same time, etc. If a governmental call to renew e-voting credentials is treated similarly to a call to mandatory jury duty, then even a long-term coercer must allow the voter to go renew occasionally, or else risk government officials coming to check on the voter and detecting the long-term coercion in that way.

G Management and Storage of Credentials

The focus of this paper is on TRIP, the e-voting registration stage in Votegral: in particular the printing of real and fake paper credentials to be activated later on any device(s) the voter trusts. Once a credential has been activated on a particular device, the long-term storage and management of that credential, potentially across several subsequent elections until the credential expires, is an important security and usability challenge beyond the scope of this paper. Nevertheless, we briefly discuss here some general considerations for the secure storage and lifecycle management of credentials.

G.1 Credential storage lifecycle

The advantage of using voting credentials that are stored on the voter's device is convenience and cost reduction, for both the registrar and the voter, compared to issuing devices, such as smart cards. However, we cannot presume that voters will keep the same device for extended periods (e.g., years), unlike other government materials such as passports. Fortunately, since the private voting materials are digitally stored on the voter's device, they can easily be transferred over to a new device using NFC or QR codes. This raises an obvious question, however: what if the voter loses their device?

Re-registration. The simplest solution is for voters to re-register for a new set of credentials. However, this approach is reasonably inconvenient for voters.

Cloud Storage. Another approach is to back up all voting data onto a cloud service [115]. Voters would then only need to authenticate with the cloud service to retrieve their voting credential. However, this method is vulnerable to single points of failures, including the potential compromise of the cloud provider or phishing attacks targeting the voter.

Threshold solutions. A third approach involves using threshold solutions via secret-sharing [128] to store their voting materials with a set of trustees: i.e., people trusted by the voter, such as friends and family. This method allows a voter to back up their credentials without depending on a single entity. To access their credential, the voter must collaborate with a threshold of their trustees to gather the necessary shares on a new device. No single trustee can act maliciously, as each trustee holds only a share of the voter's voting materials, which is completely useless in itself. However, this solution is not as convenient compared to using a cloud provider and may face reliability challenges [75, 126]. User studies on social recovery mechanisms is therefore an important necessity that could provide valuable insights into whether it would be suitable for this purpose.

G.2 Management of multiple credentials

If a voter decides to store multiple credentials on the same device, this raises the question of how the user can securely and properly access their intended credential.

Security Questions. One possible method for securing credentials is the use of security questions. A user could link each credential to a specific security question, simplifying access without the need to remember complex passwords. Alternatively, different answers to the same security question could also reveal a specific credential. However, security question-based solutions are known to be susceptible to social engineering attacks [33].

Passwords. Another approach is to link a credential with a password. While passwords are generally high-entropy alphanumeric strings, remembering multiple complex passwords can be challenging for voters and may negatively impact the usability of the system [67].

PINs. PINs are also possible but are composed of a fixed number of digits, making them lower in entropy and, therefore, less secure than passwords. Furthermore, having to remember several PINs could still be cumbersome for voters. Some applications like Signal require users to re-enter their PIN periodically to keep them from forgetting it.

H Cryptographic Primitives

This section presents the primitives used in TRIP.

Distributed Key Generation Scheme. TRIP uses a distributed key generation protocol DKG [65] that inputs a group description (G, q, g)—a cyclic group G of order q with generator g—and the number of parties n and creates a private and public key pair for each party $(P_i^{\mathsf{sk}}, P_i^{\mathsf{pk}})$ and a collective public key P^{pk}:

$$\left\{P_i^{\mathsf{sk}}, P_i^{\mathsf{pk}}\right\}, P^{\mathsf{pk}} \leftarrow \mathsf{DKG}(G, p, g, n),$$

such that $P_i^{\mathsf{pk}} = g^{P_i^{\mathsf{sk}}}$ where $P_i^{\mathsf{sk}} \leftarrow\!\!\$\ \mathbb{Z}_q$, $P^{\mathsf{pk}} = \prod_{i=1}^{n} P_i^{\mathsf{pk}}$.

ElGamal Encryption Scheme. This scheme is parameterized by a cyclic group G of prime order q and a random generator g; and consists of the following algorithms: $\mathsf{EG.KGen}(G, q, g)$ which takes as input the group definition and outputs a public key pk along with a private key sk such that $\mathsf{pk} = g^{\mathsf{sk}}$ where $\mathsf{sk} \leftarrow\!\!\$\ \mathbb{Z}_q$; a randomized encryption algorithm $\mathsf{EG.Enc}(\mathsf{pk}, m)$ which inputs a public key pk and a message $m \in G$ and outputs a ciphertext $C = (C_1, C_2) = (g^r, \mathsf{pk}^r m)$ for $r \leftarrow\!\!\$\ \mathbb{Z}_q$; and, a deterministic decryption algorithm $\mathsf{EG.Dec}(\mathsf{sk}, C)$ which takes as input a private key sk and a ciphertext C and outputs a message $m = C_2(C_1^{\mathsf{sk}})^{-1}$.

Signature Scheme. TRIP uses a EUF-CMA signature scheme defined by the following three algorithms: a randomized key generation algorithm $\mathsf{Sig.KGen}(1^\lambda)$ which takes as input the security parameter and outputs a signing key pair $(\mathsf{sk}, \mathsf{pk})$; a signature algorithm $\mathsf{Sig.Sign}(\mathsf{sk}, m)$ which inputs a private key and a message $m \in \{0,1\}^*$ and outputs a signature σ; a signature verification algorithm $\mathsf{Sig.Vf}(\mathsf{pk}, m, \sigma)$ which outputs $\top$ if σ is a valid signature of m and $\perp$ otherwise; and, an algorithm $\mathsf{Sig.PubKey}(\mathsf{sk})$ that takes as input a private key sk and outputs the corresponding public key pk.

Hash. TRIP utilizes a cryptographic secure hash function H, for which the output is 2λ, for security parameter λ.

Message Authentication Code. TRIP uses a message authentication code scheme defined by the following two algorithms: a probabilistic signing algorithm $\mathsf{MAC.Sign}(k, m)$ which takes as input a (secret) key k and a message m and outputs an authorization tag τ; and a deterministic verification algorithm $\mathsf{MAC.Vf}(k, m, \tau)$ which takes as input the (secret) key k, the message m and the authorization tag τ and outputs either $\top$ for accept or $\perp$ for reject.

Zero-Knowledge Proof of Equality. TRIP employs an interactive zero-knowledge proof of equality of discrete logarithms [39] ZKPoE so that a prover P can convince a verifier V that P knows x, given messages $y \equiv g_1^x \pmod p$ and $z \equiv g_2^x \pmod p$ without revealing x. In *interactive* zero knowledge proofs, the verifier V must provide the challenge only *after* the prover P has computed and provided the commit to V.

I Formal System and Threat Models

I.1 System Model

The Votegral architecture involves the following key actors.

Ledger. The ledger $\mathbb{L}$ is an append-only, always available, publicly accessible data structure. It includes three "sub-ledgers", the Registration Ledger $\mathbb{L}_R$, the Envelope Commitment Ledger $\mathbb{L}_E$, and the Ballot Ledger $\mathbb{L}_V$. We idealize the ledger as a tamper-evident, globally consistent view with perfect access: All actors observe the same authentic ledger state, and any tampering (e.g.,, denials, alterations, or view manipulations) is detectable with overwhelming probability. This assumption abstracts away consensus, allowing us to focus on e-voting security definitions. In practice, ledger integrity holds under honest-majority assumptions with decentralized querying and well-decentralized nodes.

Registrar. The registrar $\mathbb{R}$ enrolls eligible voters from a given electoral roll $\mathbb{V}$. The registrar consists of: (1) kiosks $\mathbb{K} = \{K_1, \ldots, K_{n_K}\}$, each in a privacy booth, which issue voters credentials; (2) envelope printers $\mathbb{P} = \{P_1, \ldots, P_{n_P}\}$, which issue the envelopes that the voters use during credentialing; and, (3) registration officials $\mathbb{O} = \{O_1, \ldots, O_{n_O}\}$, represented by their *official supporting device* (OSD), who authenticate voters and authorize their credentialing sessions.

Device / Adv.	Ledger	Authority	OSD	Kiosk	Envelope Printers	VSD w/ Cred. Real	VSD w/ Cred. Fake
Integrity	No*	All	Yes†	Yes†	Yes†	No	No
Privacy	Yes	$n_A - 1$	Yes	Yes	Yes	No	Yes
Coercion	Yes	$n_A - 1$	No	No	No	No‡	Yes

Table 1. Threat Model: This table depicts an adversary's ability to compromise a device and the entity or credential it represents.
* The ledger is treated as a trusted idealized component, akin to ideal ledgers in blockchain models for . See threat model for details.
† The risk of the adversary compromising voter registration (see Section 5), is small per registration, and becomes negligible when considering the combined probability across registrations.
‡ If voters cannot conceal their device from a coercer, they can entrust their real credential to a trusted third party to cast votes on their behalf.

Voters. Voters on the electoral roll $\mathbb{V} = \{V_1, \ldots, V_{n_V}\}$ who interact with the system to register and vote. Voters obtain their credentials in-person at the registrar, and activate them on their voter supporting device (VSD) to cast ballots on a ledger. VSDs periodically monitor the ledger to inform voters of relevant updates, such as a registration session.

Authority. The authority $\mathbb{A} = \{A_1, \ldots, A_{n_A}\}$ consists of n_A members who jointly process the ballots cast on the ledger to produce a publicly verifiable tally. The authority also manages election logistics, sets policy, and is accountable to the public.

I.2 Threat Model

We define three adversaries corresponding to specific scenarios[3]: integrity $\mathcal{I}$, privacy $\mathcal{P}$ and coercion $\mathcal{C}$. Table 1 illustrates which actors, represented by their device for added granularity, each adversary may compromise. We assume all adversaries are computationally bounded, cryptographic primitives are secure, communication channels are secure (e.g., TLS), unless stated otherwise.

Integrity. $\mathcal{I}$'s goal is to manipulate the election outcome without detection, as any detected interventions could undermine public confidence in the results. $\mathcal{I}$ cannot compromise the electoral roll nor the VSDs.[4]

Privacy. $\mathcal{P}$'s goal is to *reveal* a voter's real vote to, for example, target specific voters with personalized political advertising to influence their vote in future elections [118]. $\mathcal{P}$ can compromise all but *one* authority member.[5] $\mathcal{P}$ cannot compromise the VSDs containing voters' real credentials.

Coercion. Unlike $\mathcal{P}$, the coercion adversary $\mathcal{C}$ aims to overtly pressure voters into casting the coercer's vote. $\mathcal{C}$'s objective

[3]An availability adversary seeking to deny service is also relevant, but we assume this issue is addressed by the use of high-availability infrastructure.
[4]Works such as [25, 42] address the latterusing a cast-as-intended mechanism.
[5]This notion is common in electronic voting literature [8, 78] and in privacy-enhancing systems in general [65, 125].

Symbol	Description
G, q, g	A cyclic group G of order q with generator g
$\mathbb{A}, \mathbb{O}, \mathbb{K}, \mathbb{P}, \mathbb{V}$	Authority, Officials, Kiosks, Envelope Printers, Electoral roll
n_A, n_O, n_K, n_P	Number of Authorities, Officials, Kiosks, Envelope printers
$\mathbb{L}, \mathbb{L}_R, \mathbb{L}_E, \mathbb{L}_V$	Ledger and Registration, Envelope & Voting (sub-)ledgers
$V_{id}, c_{\text{pc}}, c_{\text{pk}}, c_{\text{sk}}$	Voter's identifier, Public Credential, Public & Private Keys
$c, \tilde{c}_{\text{pk}}, \tilde{c}_{\text{sk}}$	A credential, A fake credential's Public and Private Keys
$\mathbf{E}, n_E, n_c$	Envelope challenges and number of envelopes & credentials
τ, s_{rk}	MAC authorization tag, Official & kiosk shared secret key
OSD, VSD	Registration official and Voter's supporting devices
$t_{in}, t_{out}, q_c, q_r$	Check-In & Check-Out Tickets, Commit & Response Codes

Table 2. Scheme Notations.

```
TRIP ($\lambda$, $(G, q, g)$, $V$, $\mathbb{L}$, $\mathbb{A}$, $\mathbb{O}$, $\mathbb{K}$, $\mathbb{P}$, $\mathbf{E}$, $s_{rk}$)

1 :  $O, K \leftarrow V(\mathbb{O}, \mathbb{K})$% Voter-chosen/given official and kiosk
2 :  $t_{in} \leftarrow (V_{id}, \_) \leftarrow \mathsf{CheckIn}(O, K, s_{rk}, V_{id})$
3 :  $c \leftarrow (q_c, t_{ot}, q_r, e) \leftarrow ((c_{\text{pc}}, Y_c, \sigma_{kc}), (V_{id}, c_{\text{pc}}, K, \sigma_{k_{ot}}),$
          $(V_{id}, c_{\text{sk}}, r, K, \sigma_{kr}), (P, e, \sigma_p))$
          $\leftarrow \mathsf{RealCred}(\lambda, (G, q, g), K, \mathbb{A}_{\text{pk}}, \mathbf{E}, t_{in})$
4 :  $\mathbf{e} \leftarrow \{e\}$; $\mathbf{c} \leftarrow \{c\}$ % Used challenges & vector of credentials
5 :  for 1 to $(n_c \leftarrow V()) - 1$ do % Chosen number of envelopes
6 :    $\tilde{c} \leftarrow \mathsf{FakeCred}(\lambda, (G, q, g), K, \mathbb{A}_{\text{pk}}, \mathbf{E}_{\ominus\mathbf{e}}, t_{ot})$
7 :    $\mathbf{c} \Leftarrow \tilde{c}$; $\mathbf{e} \Leftarrow \tilde{c}[e]$
8 :  $c_v \leftarrow V(\mathbf{c})$ % Voter-chosen credential for check-out
9 :  $\mathbb{L} \leftarrow \mathsf{CheckOut}(O, \mathbb{L}, \mathbb{K}, c_v[t_{ot}])$
10 : for $i$ to $n_c$ do
11 :   $\mathbb{L} \leftarrow \mathsf{Activate}(\mathbb{L}, V, \mathbf{c}_i)$
```

Figure 6. TRIP Registration process for a prospective voter.

is to determine whether the targeted voters complied with their demands, which may involve demanding voters to cast C-dictated ballots, or reveal their real credential. C inherits the same capabilities as $\mathcal{P}$, meaning it can compromise all but one authority member, and cannot compromise the VSD containing the voter's real credential. Additionally, C cannot compromise the registrar nor observe the communication channel between the voter and the kiosk. While side-channel attacks are out-of-scope, we discuss them briefly in Appendix N. As is typical in coercion-resistant e-voting systems [41, 78], the communication channel used by voters to cast their vote is anonymous to counter the adversary's potential demand that the voter never communicates with the ledger.

J Formal TRIP Registration Protocol

This section presents TRIP, formally described in Fig. 6.

Notation. For a finite set S, $s \leftarrow\!\$\; S$ denotes that s is sampled independently and uniformly at random from S. The symbol $\ominus$ represents exclusion from a collection of elements. We denote $a \Leftarrow b$ as appending b to a, $a||b$ as concatenating b with a, $\mathbf{x}$ as a vector of elements of type x, and $\mathbf{x}[i]$ as the ith entries of the vector $\mathbf{x}$. We use $\top$ and $\bot$ to indicate success and failure, respectively. Variables used throughout the TRIP scheme are summarized in Table 2.

Primitives. TRIP requires (1) the ElGamal encryption scheme EG (2) a distributed key generation scheme DKG, (3) a EUF-CMA signature scheme Sig, (4) a secure hash function H, (5) a message authentication code scheme MAC, and (6) an interactive zero-knowledge proof of equality of discrete logarithms ZKPoE. We define these primitives in Appendix H.

J.1 Setup.

```
Setup($\lambda$, $\mathbb{V}$, $(G, p, g)$, $n_A$, $n_O$, $n_K$, $n_P$, $n_E$)

1 :  $\mathbb{L} \leftarrow \emptyset$
2 :  $\left\{A_i^{\text{sk}}, A_i^{\text{pk}}\right\}, \mathbb{A}_{\text{pk}} \leftarrow \mathsf{DKG}(G, p, g, n_A)$
3 :  $\left\{O_i, K_i, P_i \leftarrow \mathsf{Sig.KGen}(1^\lambda)\right\}_{0 \le i < n_O, 0 \le i < n_K, 0 \le i < n_P}$
4 :  $\mathbb{L}_R \leftarrow \{\mathbb{V}_i^{id}\}_{i \in \mathbb{V}}$
5 :  $\mathbf{E} \leftarrow \{P_j \leftarrow\!\$\; \mathbb{P}; e_i \leftarrow\!\$\; \mathbb{Z}_q; \mathbb{L}_E \Leftarrow (P_j^{\text{pk}}, \mathsf{H}(e_i),$
                    $\mathsf{Sig.Sign}(P_j^{\text{sk}}, \mathsf{H}(e_i)))\}_{0 \le i < n_E}$
6 :  $s_{rk} \leftarrow \{0, 1\}^\lambda$
```

Figure 7. Setup Procedure for the ledger, the authority members, the officials, the kiosks and the envelope printers with envelope issuance. The secret s_{rk} is shared between officials and kiosks.

Setup (Fig. 7) initializes the core system actors (Ledger, Authority, and Registrar). Prior works often include registration as part of a broader setup process, but we separate it to delineate registration cleanly:

- The ledger $\mathbb{L}$ becomes available and made accessible to all (including third) parties. To delineate the different types of recorded events, we introduce sub-ledgers: $\mathbb{L}_R$, $\mathbb{L}_E$, and $\mathbb{L}_V$, which are dedicated to storing registration sessions, envelope data, and votes, respectively.
- The authority members $\mathbb{A}$ run DKG and outputs a private, public keypair for each authority member $(A_i^{\text{sk}}, A_i^{\text{pk}})$ and a collective public key $\mathbb{A}_{\text{pk}}$ which is made available to all parties. $\mathbb{A}_{\text{pk}}$ must be a generator of G_q.
- Each registrar actor (OSDs, kiosks & printers) generates their own private and public key pair using $\mathsf{Sig.KGen}(1^\lambda)$. The registrar uses the electoral roll $\mathbb{V}$ to populate $\mathbb{L}_R$ with each voter's unique identifier V_i^{id}. The printer issues at least $n_E > c|\mathbb{V}| + \lambda_{\mathbf{E}}|\mathbb{K}|$ envelopes $\mathbf{E}$, where constant $c \ge 2$ represents the authority's estimate of the number of envelopes each voter consumes and $\lambda_{\mathbf{E}}$ is a security parameter detailed in Appendix L.1, which represents the minimum number of envelopes required in each

booth. If authorities underestimate the average consumption of envelopes, $\mathbb{P}$ can issue additional envelopes.[6] Each envelope contains the printer's public key P_i^{pk}, a cryptographic nonce $e \leftarrow\$ \mathbb{Z}_q$, and a signature on this nonce $\sigma_p \leftarrow \text{Sig.Sign}_p(\text{H}(e))$. For each envelope, the printer also publishes $(P_i^{\text{pk}}, \text{H}(e), \sigma_p)$ to the ledger $\mathbb{L}_E$. The officials $\mathbb{O}$ and kiosks $\mathbb{K}$ generate a shared secret key s_{rk} to create and verify MAC tags that authorize voters access to a kiosk.

J.2 Check-In.

```
CheckIn(O, K, s_rk, V_id)

     OSD(O, s_rk, V_id)                      Kiosk(K, s_rk, t_in)
1 :  $\tau_r \leftarrow \text{MAC.Sign}(s_{rk}, V_{id})$          $(V_{id}, \tau_r) \leftarrow t_{in}$
2 :  $t_{in} \leftarrow (V_{id}, \tau_r)$                  $\text{MAC.Vf}(s_{rk}, \tau_r, V_{id}) \overset{?}{=} \top$
```

Figure 8. Check-In. The offical's device issues check-in ticket and kiosk verifies authenticity of check-in ticket.

Upon successful authentication at Check-In (Fig. 8), the OSD issues the voter a check-in ticket, t_{in}, consisting of the voter's identifier, V_{id}, and an authorization tag, τ_r, on V_{id}.[7]

J.3 Authorization.

The kiosk validates the authorization tag, τ_r, when the voter presents their ticket t_{in} (Fig. 8).

J.4 Real Credential.

The kiosk now issues the voter their real credential while proving its correctness (Fig. 9a). The kiosk first generates the voter's real credential's private and public keys $(c_{\text{sk}}, c_{\text{pk}})$ and ElGamal encrypts c_{pk} using the authority's public key $\mathbb{A}_{\text{pk}}$ to obtain the voter's public credential c_{pc}. To prove that c_{pc} encrypts c_{pk} without revealing the ElGamal randomness secret x (to later enable the construction of fake credentials), the kiosk, as the prover, and the voter, as the verifier, run an interactive zero-knowledge proof of equality of discrete logarithms:[8] $\text{ZKPoE}_{C_1,X}\{(x) : C_1 = g^x \wedge X = \mathbb{A}_{\text{pk}}^x\}$. The kiosk first computes the commits $Y_1 = g^y$, and $Y_2 = \mathbb{A}_{\text{pk}}^y$ for $y \leftarrow\$ \mathbb{Z}_q$ and prints the commit q_c containing the voter's public credential c_{pc}, the commits $Y_c = (Y_1, Y_2)$, and a signature σ_{k_c} on $(V_{id} \| c_{\text{pc}} \| Y_c)$. The voter then supplies the kiosk with an envelope $E_i \leftarrow\$ \mathbf{E}$ containing the challenge e. The kiosk finally computes the response $r = y - ex$, the signatures $\sigma_{k_{ot}}$ and σ_{k_r}, and prints the check-out ticket t_{ot} and response q_r.

At this stage, the voter observes that the process adheres to the Σ-protocol sequence: commit, challenge, response. If the voter detects and reports an anomaly, we expect the registrar or some other authority to direct the voter to another kiosk and inspect the one reported, as detailed in Appendix L.3.5. The voter's device later verifies the computational correctness.

J.5 Fake Credentials.

To create a fake credential (Fig. 9b), the kiosk generates a new credential $(\tilde{c}_{\text{sk}}, \tilde{c}_{\text{pk}})$ and falsely proves that the public credential c_{pc} encrypts $\tilde{c}_{\text{pk}}$. The kiosk first derives the "new" ElGamal secret $\tilde{X} \leftarrow C_2/\tilde{c}_{\text{pk}}$. The kiosk has no knowledge of an $\tilde{x}$ that satisfies $\tilde{X} = \mathbb{A}_{\text{pk}}^{\tilde{x}}$, requiring one to solve the discrete logarithm problem. Instead, the kiosk and the voter follow an incorrect proof construction sequence that violates soundness without affecting correctness. In this sequence, the voter first supplies a new envelope $E_z \leftarrow\$ \mathbf{E}_{\ominus\mathbf{e}}$ to the kiosk, where $\mathbf{e}$ are the previously used envelopes/challenges. Then, the kiosk uses the new challenge e to compute a ZKP commit $(Y_1, Y_2) \leftarrow (g^y C_1^e, \mathbb{A}_{\text{pk}}^y \tilde{X}^e)$ for some $y \leftarrow\$ \mathbb{Z}_q$ and the ZKP response $r \leftarrow y$. The kiosk finishes by computing signatures σ_{k_c} and σ_{k_r} and printing the commit q_c, check-out t_{ot}, and response q_r sequentially, where t_{ot} is identical (both in content and visually) to the one in the real credential process. The voter can repeat this process for any number of desired fake credentials (within reasonable limits mentioned in §3.2).

J.6 Check-Out.

```
CheckOut(O, 𝕃, 𝕂, t_ot)

     OSD(O, 𝕃, 𝕂, t_ot)
1 :  $(V_{id}, c_{\text{pc}}, K^{\text{pk}}, \sigma_{k_{ot}}) \leftarrow t_{ot}$                      % Check-Out Ticket
2 :  $K^{\text{pk}} \overset{?}{\in} \mathbb{K}^{\text{pk}}$;                                   % Authorized?
3 :  $\text{Sig.Vf}(K^{\text{pk}}, \sigma_{k_{ot}}, V_{id} \| c_{\text{pc}}) \overset{?}{=} \top$            % Verify Signature
4 :  $\sigma_o \leftarrow \text{Sig.Sign}(O^{\text{sk}}, V_{id} \| c_{\text{pc}} \| \sigma_{k_{ot}})$          % Official Approval
5 :  $\mathbb{L}_R[V_{id}] \leftarrow (c_{\text{pc}}, K^{\text{pk}}, \sigma_{k_{ot}}, O^{\text{pk}}, \sigma_o)$   % Update Ledger on $V_{id}$
     VSD(𝕃, V)
6 :  $\text{Notify}(V_{id})$                                          % Notify Voter
```

Figure 10. Check-Out. The official approves the voter's registration session, publishing on the ledger their signature, the kiosk's signature and the voter's public credential. The ledger verifies signatures and the VSD monitoring the ledger notifies the voter.

At check-out (Fig. 10), the official uses their OSD scans the voter's chosen credential. The voter shows this credential in the transport state (Fig. 2c), which reveals the contents of the check-out ticket t_{ot}.

[6] Unlike paper ballots, envelopes do not expire, allowing unused ones to be saved for future registrations.

[7] For usability, as discussed in §7.5, we use a barcode instead of a QR code, and due to storage constraints in a barcode, we use MAC instead of Sig.

[8] For cryptographic purposes, we equate the voter with the verifier. However, in reality, the voter only observes the order in which QR codes are printed on the receipt without needing to understand them. The voter's device is responsible for checking the actual proof transcripts.

```
RealCred($\lambda$, $(G, q, g)$, $K$, $\mathbb{A}_{\mathrm{pk}}$, $\mathbf{E}$, $t_{in}$)

Kiosk($\lambda$, $(G, q, g)$, $K$, $\mathbb{A}_{\mathrm{pk}}$, $\mathbf{E}$, $t_{in}$)                                        Voter
 1 : $(V_{id}, \_) \leftarrow t_{in}$                                  % Check-In Ticket
 2 : $(c_{\mathrm{sk}}, c_{\mathrm{pk}}) \leftarrow \mathsf{Sig.KGen}(1^{\lambda})$                  % Real Credential
 3 : $x \leftarrow\!\!\$\ \mathbb{Z}_q$; $X \leftarrow \mathbb{A}_{\mathrm{pk}}^{x}$                                % ElGamal secret
 4 : $c_{\mathrm{pc}} \leftarrow (C_1, C_2) \leftarrow (g^x, c_{\mathrm{pk}}X)$                      % Public Credential
 5 : $y \leftarrow\!\!\$\ \mathbb{Z}_q$; $Y_c \leftarrow (Y_1, Y_2) \leftarrow (g^y, \mathbb{A}_{\mathrm{pk}}^{y})$     % ZKP commit
 6 : $\sigma_{k_c} \leftarrow \mathsf{Sig.Sign}(K^{\mathrm{sk}}, V_{id}\|c_{\mathrm{pc}}\|Y_c)$             % Commit Signature
 7 : $q_c \leftarrow (V_{id}, c_{\mathrm{pc}}, Y_c, \sigma_{k_c})$                           % Commit
 8 :                                                 $\xrightarrow{q_c}$
 9 :                                                                 $E_i \leftarrow\!\!\$\ \mathbf{E}$
10 :                                                 $\xleftarrow{E_i}$
11 : $(P^{\mathrm{pk}}, e, \sigma_p) \leftarrow E_i$                               % ZKP challenge
12 : $r \leftarrow y - ex$                                       % ZKP response
13 : $\sigma_{k_{ot}} \leftarrow \mathsf{Sig.Sign}(K^{\mathrm{sk}}, V_{id}\|c_{\mathrm{pc}})$                 % Check-Out Signature
14 : $\sigma_{k_r} \leftarrow \mathsf{Sig.Sign}(K^{\mathrm{sk}}, c_{\mathrm{pk}}\|\mathsf{H}(e\|r))$
15 : $t_{ot} \leftarrow (V_{id}, c_{\mathrm{pc}}, K^{\mathrm{pk}}, \sigma_{k_{ot}})$                       % Check-Out Ticket
16 : $q_r \leftarrow (c_{\mathrm{sk}}, r, K^{\mathrm{pk}}, \sigma_{k_r})$                          % Response
17 :                                                 $\xrightarrow{t_{ot}}$
18 :                                                 $\xrightarrow{q_r}$
```

(a) Real Credential Creation Process. Voter and kiosk follow a sound zero-knowledge proof construction.

```
FakeCred($\lambda$, $(G, q, g)$, $K$, $\mathbb{A}_{\mathrm{pk}}$, $\mathbf{E}_{\ominus\mathbf{e}}$, $t_{ot}$)

Kiosk($\lambda$, $(G, q, g)$, $K$, $\mathbb{A}_{\mathrm{pk}}$, $\mathbf{E}_{\ominus\mathbf{e}}$, $t_{ot}$)                                      Voter
 1 : $(V_{id}, c_{\mathrm{pc}}, \_, \_) \leftarrow t_{ot}$                            % Unpack Check-out Ticket
 2 : $(\tilde{c}_{\mathrm{sk}}, \tilde{c}_{\mathrm{pk}}) \leftarrow \mathsf{Sig.KGen}(1^{\lambda})$                   % Fake Credential
 3 : $(C_1, C_2) \leftarrow c_{\mathrm{pc}}$                                  % Unpack Public Credential
 4 : $\tilde{X} \leftarrow C_2 / c_{\mathrm{pk}}$                                   % Derive ElGamal secret
 5 :                                                                 $E_z \leftarrow\!\!\$\ \mathbf{E}_{\ominus\mathbf{e}}$
 6 :                                                 $\xleftarrow{E_z}$
 7 : $(P^{\mathrm{pk}}, e, \sigma_p) \leftarrow E_z$                               % ZKP challenge
 8 : $y \leftarrow\!\!\$\ \mathbb{Z}_q$                                         % ZKP commit
 9 : $Y_c \leftarrow (Y_1, Y_2) \leftarrow (g^y C_1^e, \mathbb{A}_{\mathrm{pk}}^{y}\tilde{X}^e)$        % ZKP commit
10 : $r \leftarrow y$                                           % ZKP response
11 : $\sigma_{k_c} \leftarrow \mathsf{Sig.Sign}(K^{\mathrm{sk}}, V_{id}\|c_{\mathrm{pc}}\|Y_c)$
12 : $\sigma_{k_r} \leftarrow \mathsf{Sig.Sign}(K^{\mathrm{sk}}, \tilde{c}_{\mathrm{pk}}\|\mathsf{H}(e\|r))$
13 : $q_c \leftarrow (V_{id}, c_{\mathrm{pc}}, Y_c, \sigma_{k_c})$
14 : $q_r \leftarrow (\tilde{c}_{\mathrm{sk}}, r, K^{\mathrm{pk}}, \sigma_{k_r})$
15 :                                                 $\xrightarrow{q_c}$
16 :                                                 $\xrightarrow{t_{ot}}$
17 :                                                 $\xrightarrow{q_r}$
```

(b) Fake Credential Creation Process. Voter and kiosk follow an unsound zero-knowledge proof construction. Envelopes cannot be reused.

Figure 9. Voting Credential Creation Process.

The OSD first checks the credential's authenticity by checking the kiosk's public key $K^{\mathrm{pk}} \in \mathbb{K}^{\mathrm{pk}}$, and verifying the signature $\sigma_{k_{ot}}$. OSD then provides their stamp of approval with a digital signature σ_o on the voter's identifier V_{id}, the voter's public credential c_{pc} and the kiosk's signature $\sigma_{k_{ot}}$. Finally, OSD updates the ledger entry V_{id} with $(c_{\mathrm{pc}}, K^{\mathrm{pk}}, \sigma_{k_{ot}}, O^{\mathrm{pk}}, \sigma_o)$. Once updated, the ledger $\mathbb{L}$ performs the necessary checks and VSD notifies the voter about their recent registration with information on how to report any irregularities.

J.7 Activation.

During activation (Fig. 11), the voter uses their VSD to scan a credential in the activate state (Fig. 2d). This reveals the commit q_c, envelope e, and response q_r; the check-out ticket t_{ot} is not visible. VSD then verifies the integrity of the credential by (1) verifying the signatures $(\sigma_{k_c}, \sigma_{k_r}, \sigma_p)$, (2) deriving the ElGamal secret X and verifying the ZKP, (3) checking whether the public credential c_{pc} matches the public credential on the ledger c'_{pc}, and (4) checking that the challenge e has not been used via the ledger $\mathbb{L}_E$. Upon success, the device publishes the challenge e on $\mathbb{L}_E$ and stores the credential c_{sk} for future voting. The device publishes the envelope challenge on the ledger for integrity, ensuring that the challenges are unique. Upon failure, VSD reports the offending actor that results from the failure check, and instructs the voter to re-register.

We present the voting and tallying scheme in Appendix K.

```
Activate($\mathbb{L}$, $V$, $c$)

VSD($(G, q, g)$, $\mathbb{L}$, $V$, $c$)
 1 : $((V'_{id}, c_{\mathrm{pc}}, Y_c, \sigma_{k_c}), \_, (c_{\mathrm{sk}}, r, K^{\mathrm{pk}}, \sigma_{k_r}),$
       $(P^{\mathrm{pk}}, e, \sigma_p)) \leftarrow (q_c, \_, q_r, e) \leftarrow c$                 % Unpack Credential
 2 : $c_{\mathrm{pk}} \leftarrow \mathsf{Sig.PubKey}(c_{\mathrm{sk}})$                               % Get Public Key
 3 : $\mathsf{Sig.Vf}(K^{\mathrm{pk}}, \sigma_{k_c}, V'_{id}\|c_{\mathrm{pc}}\|Y_c) \stackrel{?}{=} \top$                  % Receipt Integrity Check 1
 4 : $\mathsf{Sig.Vf}(K^{\mathrm{pk}}, \sigma_{k_r}, c_{\mathrm{pk}}\|\mathsf{H}(e\|r)) \stackrel{?}{=} \top$                 % Receipt Integrity Check 2
 5 : $\mathsf{Sig.Vf}(P^{\mathrm{pk}}, \sigma_p, H(e)) \stackrel{?}{=} \top$                          % Envelope Integrity Check
 6 : $(C_1, C_2) \leftarrow c_{\mathrm{pc}}$; $X \leftarrow C_2 / c_{\mathrm{pk}}$                        % Derive ElGamal Secret
 7 : $(Y_1, Y_2) \leftarrow Y_c$                                      % Extract ZKP commitments
 8 : $Y_1 \stackrel{?}{=} g^r C_1^e$; $Y_2 \stackrel{?}{=} A^r X^e$                               % Verify ZKP
 9 : $(c'_{\mathrm{pc}}, K'^{\mathrm{pk}}, \sigma_{k_{ot}}, O^{\mathrm{pk}}, \sigma_o) \leftarrow \mathbb{L}_R[V_{id}]$            % Voter Reg. Session
10 : $c'_{\mathrm{pc}} \stackrel{?}{=} c_{\mathrm{pc}} \land K'^{\mathrm{pk}} \stackrel{?}{=} K^{\mathrm{pk}} \land V'_{id} \stackrel{?}{=} V_{id}$         % Verify Public Cred & Actors
11 : $e \stackrel{?}{\notin} \mathbb{L}_E[\mathsf{H}(e)]$;   $\mathbb{L}_E[\mathsf{H}(e)] \leftarrow e$               % Challenge Unused & Append
```

Figure 11. Credential Activation. Verifies the integrity of the credential: if any procedure with fails, then the process aborts. Upon success, VSD stores the credential's private key c_{sk}.

K Formal Voting and Tallying Protocol

In this section, we present the voting and a linear time tallying scheme in the broader Votegral scheme.

K.1 Overview

The challenge with JCJ's scheme is that submitted ballots are not authenticated, allowing anyone, including third party actors, to cast ballots. Although these ballots are removed as fake ballots during tallying, the filtering process is prohibitive, with a complexity of $O(|B| \cdot n_V)$ where $|B|$ is the number of cast ballots and n_V is the number of registered voters. This complexity is at least quadratic when each voter casts a vote, i.e., $|B| = n_V$.

One approach to tally votes in linear time is presented by Weber et al. [136]. Instead of performing plaintext equivalence tests between each entry in B and C, the election authority calculates a deterministic tag for each entry in B and C and then matches the tags. This method reduces the complexity to $O(|B| + n_V)$. However, this approach was later found to be vulnerable to coercion, as a coercer could cast a ballot with a specially crafted fake credential to determine whether the voter complied with their demands. Therefore, to address this issue in Votegral, we can leverage TRIP to ensure ballots are cast using only authorized credentials—both real and fake—that are created during registration. For instance, the kiosk can provide a digital signature on each credential c_{pk} it issues, which must be present when casting a vote.

K.2 Voting Scheme

```
Vote($\mathbb{L}, c, \mathbb{A}_{pk}, \beta, M$)

    VSD($(G, q, g), V_{id}, \mathbb{L}, c, \beta$)
1 : $c'_{\text{pc}} \leftarrow L_R[V_{id}]; c'_{\text{pc}} \overset{?}{=} c_{\text{pc}}$                          % Current?
2 : $E_{vt} \leftarrow \mathsf{EG.Enc}(\mathbb{A}_{pk}, \beta)$                                 % Encrypt candidate choice
3 : $E_{cr} \leftarrow \mathsf{EG.Enc}(\mathbb{A}_{pk}, c_{pk})$                                % Encrypt credential
4 : $\Pi_\beta \leftarrow \mathsf{NIZK}(\beta \in M)$                                          % Proof of valid candidate choice
5 : $\Pi_\sigma \leftarrow \mathsf{NIZK}(E_{cr} \text{ encrypts } c_{pk})$                        % Proof of credential
6 : $\sigma_{vt} \leftarrow \mathsf{Sig.Sign}(c_{sk}, (E_{vt}, E_{cr}, \Pi_\beta, \Pi_\sigma))$   % Sign the ballot
7 : $B \leftarrow (c_{\text{pk}}, E_{vt}, E_{cr}, \Pi_\beta, \Pi_\sigma, \sigma_{vt})$           % Store Ballot
8 : $L_V \leftarrow B$                                                          % Post ballot on ledger
```

Figure 12. Voting Voter uses the authorities' public key and its real credential's private signing key to cast its ballot.

When casting a vote (Fig. 12), VSD first confirms that the public credential on the device still remains the voter's public credential on the ledger, verifying that the credential is part of the voter's latest registration session. If successful, it begins the voting process by ElGamal encrypting the voter's chosen vote along with the credential selected by the voter. Furthermore, VSD constructs NIZK proofs to prove that the voter's choice is among the candidates. Additionally, VSD digitally signs the ballot with the credential's private key and proves with a NIZK that the associated public key is the same. A ballot thus consists of two ElGamal ciphertexts—one an encrypted credential and the other an encrypted vote—a digital signature and NIZK proofs. When the ledger receives this ballot, it verifies that the credential's public key was issued by the kiosk, the digital signature is valid and the NIZK proofs are valid.

K.3 Tally.

We adopt the Tally operation from the TRIP API (Fig. 13) but for simplicity, we focus exclusively on producing the tally X and simply mention when proofs are generated. The resulting proof P is simply a concatenation of these proofs.

$\mathbb{A}$ first collects each voter's public credential c_{pc} from the roster $\mathbb{L}_R$ as $\mathbf{C}^r$, and then collects each encrypted ballot cast from $\mathbb{L}_V$ as $(\mathbf{C}^v, \mathbf{V})$. As presented in L.1, ballots in $\mathbb{L}_V$ contain two ElGamal ciphertexts, one for the credential and one for the vote.

$\mathbb{A}$ then runs Neff's shuffling protocol [102] to shuffle $\mathbf{C}^r$ as $\mathbf{C}^r_s$ under some permutation π_1 along with a proof for correctness. $\mathbb{A}$ similarly shuffles $(\mathbf{C}^v, \mathbf{V})$ as $(\mathbf{C}^v_s, \mathbf{V}_s)$ but under the same permutation π_2 to preserve association.

$$\mathbf{C}^r_s = \pi_1(\mathbf{C}^r), \tag{1}$$
$$\mathbf{C}^v_s = \pi_2(\mathbf{C}^v), \tag{2}$$
$$\mathbf{V}_s = \pi_2(\mathbf{V}). \tag{3}$$

Neff describes the latter as a K-shuffle and provides the necessary modifications to the protocol to compute a corresponding proof.

$\mathbb{A}$ then runs a distributed deterministic-tag protocol [65, 136] first on $\mathbf{C}^r_s$ and then on $\mathbf{C}^v_s$, creating a deterministic tag on each ciphertext that relates to the message m without ever revealing m along with a proof for correctness. m in TRIP is the public key of the credential: c_{pk} (line 4, Fig. 9a, §4). From the resulting sets $\mathbf{C}^r_t$ and $\mathbf{C}^v_t$, $\mathbb{A}$ can collect the set of encrypted real votes from the tags $\mathbf{C}^t_{v_i} \in \mathbf{C}^t_r$:

$$\mathbf{V}_r = \{\mathbf{V}_{s_i} | \mathbf{C}^t_{v_i} \in \mathbf{C}^t_r\}$$

Finally, A verifiably decrypts $\mathbf{V}_r$ as $\mathbf{V}^d_r$ using distributed ElGamal decryption protocol which also generates zero-knowledge proofs that vote $\mathbf{v}_i$ is the correct decryption of $\mathbf{V}^d_{r_i}$.

L Detailed Security Analysis and Formal Proofs

We formally prove that TRIP satisfies coercion-resistance and individual verifiability, and provide a proof sketch for the privacy adversary. Tables 2 and 3 show our notation and summarize our variables.

TRIP API. We first redefine the TRIP API (Fig. 13), where algorithms append to a ledger instead of submitting to it. Since the registrar is either all malicious (integrity, privacy) or all honest for coercion, we denote $\mathbb{R}$ to represent the kiosks, registration officials and the envelope printers.

- $c^i_{\text{sk}}, c^i_{\text{pc}}, P^i, \mathbb{L}_R \leftarrow \mathsf{RealCred}(\mathbb{L}_R, \mathbb{R}_{\text{sk}}, \mathbb{V}^{id}_i, \lambda)$: takes as input the ledger $\mathbb{L}_R$, the registrars' private key $\mathbb{R}_{\text{sk}}$, a voter's identifier $\mathbb{V}^{id}_i$ and a security parameter λ and outputs the voter's private and public credential c^i_{sk} and c^i_{pc}, correctness proofs P^i and an updated registration ledger $\mathbb{L}_R$. For simplicity, RealCred incorporates the CheckIn and CheckOut processes required in TRIP.
- $\mathbf{c}^i_f, \mathbf{P}^i \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\text{sk}}, \mathbb{V}^{id}_i, c^i_{\text{pc}}, n_f, \lambda)$: takes as input the registrar's private key R_{sk}, the voter's identifier $\mathbb{V}^{id}_i$, the voter's public credential c^i_{pc}, the number of fake credentials $n_f \in \mathbb{N}$, and a security parameter λ and outputs n_f fake credentials $\mathbf{c}^i_f$ and n_f proofs of correctness $\mathbf{P}^i$.
- $out, \mathbb{L}_R \leftarrow \mathsf{Activate}(\mathbb{L}_R, c, P)$: takes as input the registration ledger $\mathbb{L}_R$, a credential c and the credential's correctness proof P and outputs $out \in \{\top, \bot\}$ and an updated registration ledger $\mathbb{L}_R$.
- $\mathbb{L}_V \leftarrow \mathsf{Vote}(\mathbf{c}_{\text{sk}}, T_{\text{pk}}, n_M, D_{n_U, n_C}, \lambda)$: takes as input the set of credentials $\mathbf{c}_{\text{sk}}$, the talliers' public key T_{pk}, the candidate list n_M, and a probability distribution D_{n_H, n_M} over the possible (voter, candidate) pairs[a]. It appends a TRIP-formatted ballot for each credential in $\mathbf{c}_{\text{sk}}$ to the ledger $\mathbb{L}_V$.
- $(X, P) \leftarrow \mathsf{Tally}(T_{\text{sk}}, \mathbb{L}, n_M, \lambda)$: takes as input the talliers' private key T_{sk}, the ledger $\mathbb{L}$, the candidates n_M, and a security parameter λ and outputs the tally X and a proof P showing that the talliers computed the tally correctly.
- $(X, P) \leftarrow \mathsf{Ideal\text{-}Tally}(T_{\text{sk}}, \mathbb{L}, n_M, \lambda)$: takes as input the talliers' private key T_{sk}, the ledger $\mathbb{L}$, the candidates n_M, and a security parameter λ and outputs an ideal-tally X and a proof P. This algorithm, as defined in JCJ and only used in the ideal game, differs from Tally by not counting any ballots cast by the adversary if the bit $b = 0$.

[a]This distribution captures the uncertainty of honest voters' choices, which impairs the adversary's ability to detect coercion.

Figure 13. TRIP API.

Notation	*Description*
$\mathbb{R}$	Registrar (combines kiosks, envelope printers and officials)
$\mathbb{L}_V, X, P, \text{st}$	Voting (sub-)ledger, Tally & Tally Proofs, (Adversarial) state
D^c, D^v	Probability distribution of fake credentials and votes
β, n^c_j, M	Target Vote, Target number of total credentials, Voting options
$C, \mathbf{V}_C, j, \beta$	Coercer, C-controlled voters, C-target voter, C-intended ballot
n_V, n_C, n_M	Number of voters, controlled voters and voting options

Table 3. Proof Notations.

L.1 Coercion-Resistance

We prove that TRIP is coercion-resistant by showing that the difference between C's winning probability in a real game – representing the adversary's interactions with our system – and in an ideal game – representing the desired level of coercion-resistance – is negligible. In both games, C's goal is to determine whether the targeted voter evaded coercion and cast a real vote. Like the total number of votes cast in an election, we treat the total number of credentials created as public information: C could trivially win if it knew exactly how many (fake) credentials *all other* voters created. The adversary's uncertainty about the target voter thus derives from the other honest voters, each of whom creates an unknown (to the adversary C) number of fake credentials, which we model as a probability distribution[9] D^c.

[9]In practice, to artificially increase this uncertainty, envelope printers can post challenges on the ledger *without* printing a corresponding envelope and gradually release these values, similar to the JCJ option of voting authorities or third parties intentionally injecting fake votes to add noise.

Game C-Resist-TRIP-Ideal$^{C',b}(\lambda, \mathbb{V}, \mathbb{R}, \mathbb{A}, M, n_C)$

```
1 :  $\mathbf{V}_C, \{n^i_c\}_{1 \le i \le |\mathbf{V}_C|}, \text{st}_{C'} \leftarrow C'(\mathbb{V}, \text{“Choose controlled voter set”})$
2 :  $\mathbb{L}^{JCJ}, \mathbb{L}_E, n_V, n_M, n'_C \leftarrow \emptyset, \emptyset, |\mathbb{V}|, |M|, |\mathbf{V}_C|$
3 :  $(j, \beta, \text{st}_{C'}) \leftarrow C'(\text{st}_{C'}, \text{“Pick target voter and their parameters”})$
4 :  **if** $n'_C \neq n_C$ or $j \notin \{1, 2, ..., n_V\} \setminus \mathbf{V}_C$ **then abort**
5 :  **for** $i = 1$ **to** $n_V$ **do** % *All voters — real credential*
6 :      $c^i_{\text{sk}}, c^i_{\text{pc}}, \_, \mathbb{L}^{JCJ}_R \leftarrow \mathsf{RealCred}(\mathbb{L}^{JCJ}_R, \mathbb{R}_{\text{sk}}, \mathbb{V}^i_{id}, \lambda)$
7 :      **if** $i \in \mathbb{V} \setminus (\mathbf{V}_C \cup \{j\})$ **do** % *Honest voters — fake credentials*
8 :          $\tilde{\mathbf{c}}^i_{\text{sk}}, \_ \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\text{sk}}, \mathbb{V}^i_{id}, c^i_{\text{pc}}, D^c_{1,[0,\text{inf}]}, \lambda)$
9 :          $\_, \mathbb{L}_E \Leftarrow \left\{\mathsf{Activate}(\mathbb{L}_E, \tilde{c}^i_{\text{sk}})\right\}_{1 \le i \le |\tilde{\mathbf{c}}|}$
10 : **if** $b = 0$ **then** % *Target voter evades — casts real vote*
11 :     $\tilde{c}^j_{\text{sk}}, \_ \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\text{sk}}, \mathbb{V}^j_{id}, c^j_{\text{pc}}, 1, \lambda)$
12 :     $\_, \mathbb{L}_E \Leftarrow \left\{\mathsf{Activate}(\mathbb{L}_E, c^j_{\text{sk}}), \mathsf{Activate}(\mathbb{L}_E, \tilde{c}^j_{\text{sk}})\right\}$
13 :     $\mathbb{L}^{JCJ}_V \Leftarrow \mathsf{Vote}(c^j_{\text{sk}}, \mathbb{A}_{\text{pk}}, M, \beta, \lambda)$
14 : $c_{C'} \leftarrow c^j_{\text{sk}}$ % *Target voter always releases their real credential*
15 : $\mathbb{L}^{JCJ}_V \Leftarrow \left\{\mathsf{Vote}(c^i_{\text{sk}}, \mathbb{A}_{\text{pk}}, M, D^v_{n_V - n_C, n_M}, \lambda)\right\}_{i \in \mathbb{V} \setminus \{\mathbf{V}_C \cup j\}}$
16 : $\mathbb{L}^{JCJ}_V \Leftarrow C'(\text{st}_{C'}, \mathbb{L}^{JCJ}_R, c_{C'}, \left\{c^i_{\text{sk}}\right\}_{i \in \mathbf{V}_C}, \lambda, \text{“Coercer casts ballots”})$
17 : $(\mathbf{X}, \_) \leftarrow \mathsf{Ideal\text{-}Tally}(\mathbb{A}_{\text{sk}}, \mathbb{L}^{JCJ}_V, M, \mathbb{L}^{JCJ}_R, \lambda)$
18 : $b' \leftarrow C'(\text{st}_{C'}, \mathbf{X}, \mathbb{L}_E, \text{“Guess } b\text{”})$
19 : **return** $b' = b$
```

Figure 14. Game C-Resist-TRIP-Ideal. The TRIP ideal game for coercion-resistance, adapted from JCJ, that considers the adversary's probabilistic knowledge of honest voters' fake credentials. Notations are defined in Tables 2 and 3, TRIP API in Fig. 13.

To achieve statistical uncertainty also on the voting choice, we adopt the same approach for the content of the ballot with the distribution D^v. This "anonymity among the honest voters" mimic the reasoning by which votes themselves are considered to be (statistically) protected once anonymized. We present the ideal game in Fig. 14 to highlight TRIP's level of coercion-resistance and defer the proof to Appendix L.1.

In this ideal game, adapted from JCJ [78], the coercer chooses the target voter j and $n_C < n_V$ controlled voters who abide by the coercer's demands. All voters obtain their real credential while honest voters also create and activate fake ones. The envelope ledger $\mathbb{L}_E$'s content (line 18) releases to C the number of total credentials created during Activate. The distribution D^c reflects the statistical uncertainty pertaining to the number of fake credentials created and activated by honest voters, which could potentially be zero. If the challenge bit $b = 0$, the voter uses the real credential to cast a vote. The challenger then gives C the real credential of each controlled voter and the target voter. The honest voters and the coercer then proceed to cast ballots, where honest votes—and thus the adversarial statistical uncertainty—are drawn from the distribution D^v. The degree of coercion-resistance is thus bounded by the uncertainty the adversary faces about honest voters' behaviors. For tallying, we use Ideal-Tally, an

Game $\text{C-Resist}^{C,b}(\lambda, \mathbb{V}, \mathbb{R}, \mathbb{A}, M, n_C)$

```
1 :  $\mathbf{V}_C, \left\{n_f^i\right\}_{i \in \mathbf{V}_C}, \mathrm{st}_C \leftarrow C(\mathbb{V}, \text{“Choose controlled voter set”})$
2 :  $\mathbb{L}, n_V, n_M, n'_C \leftarrow \emptyset, |\mathbb{V}|, |M|, |\mathbf{V}_C|$
3 :  $(j, n_f^j, \beta, \mathrm{st}_C) \leftarrow C(\mathrm{st}_C, \text{“Pick target voter and their parameters”}$
4 :  **if** $n'_C \neq n_C$ or $j \notin \{1, 2, ..., n_V\} \setminus \mathbf{V}_C$ **then abort endif**
5 :  **for** $i = 1$ **to** $n_V$ **do**
6 :     $c_{\mathsf{sk}}^i, c_{\mathsf{pc}}^i, P_c^i, \mathbb{L}_R \leftarrow \mathsf{RealCred}(\mathbb{L}_R, \mathbb{R}_{\mathsf{sk}}, V_i^{id}, \lambda)$
7 :     **if** $i = j$ and $b = 0$ **then**  % *Target voter evades coercion*
8 :        $\tilde{\mathbf{c}}_{\mathsf{sk}}^j, \tilde{\mathbf{P}}_c^j \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\mathsf{sk}}, \mathbb{V}_{id}^j, c_{\mathsf{pc}}^j, n_f^j + 1, \lambda)$
9 :     **elseif** $i \in (\mathbf{V}_C \cup j)$ **then**  % *C-Controlled Voters; target voter submits*
10 :        $\tilde{\mathbf{c}}_{\mathsf{sk}}^i, \tilde{\mathbf{P}}_c^i \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\mathsf{sk}}, \mathbb{V}_{id}^i, c_{\mathsf{pc}}^i, n_f^i, \lambda)$
11 :     **else**  % *Honest voters*
12 :        $\tilde{\mathbf{c}}_{\mathsf{sk}}^i, \tilde{\mathbf{P}}_c^i \leftarrow \mathsf{FakeCreds}(\mathbb{R}_{\mathsf{sk}}, \mathbb{V}_{id}^i, c_{\mathsf{pc}}^i, D_{1,[0,\inf]}^c, \lambda)$
13 :     **endif**
14 :     $\mathbf{c}^i \leftarrow (\tilde{\mathbf{c}}_{\mathsf{sk}}^i, \tilde{\mathbf{P}}_c^i);\ \mathbf{c}^i \Leftarrow (c_{\mathsf{sk}}^i, P_c^i)$
15 :  **endfor**
16 :  $\mathbf{c}_C \leftarrow \mathbf{c}^i$ % *Target voter releases all credentials*
17 :  **if** $b = 0$ **then**  % *Target voter evades coercion*
18 :     $\_, \mathbb{L}_E \leftarrow \mathsf{Activate}(\mathbb{L}_E, c_{\mathsf{sk}}^j)$
19 :     $\mathbb{L}_V \Leftarrow \mathsf{Vote}(c_{\mathsf{sk}}^j, \mathbb{A}_{\mathsf{pk}}, M, \beta, \lambda)$
20 :     $\mathbf{c}_C \leftarrow (\tilde{\mathbf{c}}^j, \tilde{\mathbf{P}}^j)$ % *Target voter releases only fake credentials*
21 :  **endif**
22 :  **for** $i \in \mathbb{V} \setminus (\mathbf{V}_C \cup \{j\})$ **do** % *Honest voters cast vote*
23 :     $\_, \mathbb{L}_R \leftarrow \left\{\mathsf{Activate}(\mathbb{L}_R, c_{\mathsf{sk}}^i)\right\}_{1 \leq i \leq |\mathbf{c}^i|}$
24 :     $\mathbb{L}_V \Leftarrow \left\{\mathsf{Vote}(c_{\mathsf{sk}}^i, \mathbb{A}_{\mathsf{pk}}, M, D_{n_V - n_C, n_M}^v, \lambda)\right\}_{1 \leq i \leq |\mathbf{c}^i|}$
25 :  **endfor**
26 :  $\mathbb{L} \Leftarrow C(\mathrm{st}_C, \mathbb{L}, \mathbf{c}_C, \{\mathbf{c}^i\}_{i \in \mathbf{V}_C}, \lambda, \text{“Activate and cast votes”})$
27 :  $(\mathbf{X}, P_t) \leftarrow \mathsf{Tally}(\mathbb{A}_{\mathsf{sk}}, \mathbb{L}, n_C, \lambda)$
28 :  $b' \leftarrow C(\mathrm{st}_C, \mathbf{X}, P_t, \mathbb{L}, \text{“guess } b\text{”})$
29 :  **return** $b' = b$
```

Figure 15. Game C-Resist.

algorithm adopted from JCJ [78], which tallies all votes from honest and controlled voters, but only tally the adversary's vote using the voter's real credential if $b = 1$. Given the tally along with the previously given credentials, the adversary guesses the bit b, i.e., whether the victim voter has cast a ballot.

Theorem 1 (Coercion-resistance, informal)**.** *The TRIP registration scheme (within the JCJ remote electronic voting scheme*[10] *[78]) is coercion-resistant under the decisional-Diffie-Hellman assumption in the random oracle model.*

Definition 2 (Coercion-resistance)**.** *A scheme is coercion-resistant if for all PPT adversaries C, all security parameters $\lambda \in \mathbb{N}$, and all parameters $\mathbb{V}, \mathbb{R}, \mathbb{A}, M, n_C$, the following holds:*

$$\mathsf{Adv}_{C,\text{C-Resist}}^{coer}(\lambda, \cdot) =$$
$$\left| \Pr_{C,b}[\text{C-Resist}^{C,b}(\cdot) = 1] - \Pr_{C',b}[\text{C-Resist-Ideal}^{C',b}(\cdot) = 1] \right|$$
$$\leq \mathsf{negl}(\lambda),$$

where the probability is computed over all the random coins used by the algorithms in the scheme.

We introduce the following three major changes from JCJ games to model TRIP's behavior.

Change #1: Voter Registration Algorithms. To model information beyond the credential (e.g., proofs of correctness), we replace JCJ algorithms register and fakekey with RealCred and FakeCreds, respectively. We also employ Activate to model additional registration data on the ledger (e.g., envelope challenges). In our C-Resist games, Activate consistently returns *out* as $\top$ since the registrar $\mathbb{R}$ is trusted.

Change #2: Fake Credentials Issuance. In contrast to the JCJ game where C has no influence over the voter before the voting phase, TRIP allows C to interact with voters before registration. Specifically, C can demand voters to generate $n_f \in \mathbb{N}$ fake credentials during registration. We use the probability distribution D^c to model the uncertainty around fake credentials created and activated by honest voters.

The coercion adversary C has no control on the probability distribution D^c and we enforce this in the system through the parameter λ_E. This security parameter represents the minimum number of envelopes that each booth requires to ensure that coerced voters cannot accurately count all the envelopes that a booth contains. This system property—that is, a minimum number of envelopes per booth—ensures in practice that the distribution D^c is out of the coercion adversary's control, as we assume for the rest of the proof.

Change #3: Ledger Entries. While JCJ uses a trusted registration algorithm to generates a voting roster with each voter's public credential c_{pc}, TRIP incorporates a more complex protocol to achieve individual verifiability. Specifically, during Activate, the envelope challenge is disclosed on the envelope ledger, permitting the adversary to discern the total number of fake credentials created. Additionally, the registration ledger includes digital signatures from both the kiosk and officials responsible for issuing the voter's credential. Despite these additions, we show that C's winning probability is negligibly affected, outside of the added probability distribution of fake credentials represented in the ideal game. Similar to JCJ, the winning probability of the ideal game is $\gg 0$ as coercion-resistance is bounded by the adversary's uncertainty over the behavior of honest voters.

We present our formal definition for coercion-resistance under Definition 2, and use the ideal (C-Resist-Ideal) and real (C-Resist) games in figures 14 and 15, respectively. The coercer wins if they can correctly guess the bit b, representing whether or not the targeted voter gives in to coercion. While C-Resist represents the coercive adversary C, to prove security we must compare C with another adversary C' who plays C-Resist-Ideal, which embodies the security we want to achieve against coercion. We show that the difference between the real and ideal games is negligible.

[10]This proof includes voting and tallying functions as defined in JCJ to demonstrate the complete process is coercion-resistant.

Proof. We use three hybrid games to transition from the real to the ideal game, with each game involving a protocol change.

1. **Eliminate Voting Ledger View:** Eliminate C's access to the TRIP voting ledger $\mathbb{L}_V$, as C is incapable of differentiating between a ledger filled with honest voter ballots and a randomly generated set of ballots, assuming the decisional-Diffie-Hellman assumption holds.
2. **Number of Fake Credentials:** C's ability to demand voters to create a specific number of fake credentials is equal to that of an adversary C' who *cannot* demand voters to create a specific number of fake credentials, given the distribution of the honest voters' fake credentials.
3. **Eliminate Registration Ledger View**: Eliminate C's access to TRIP roster $\mathbb{L}_R$ by introducing a new ledger $\mathbb{L}_R^{JCJ}$ where we can decouple V_i^{id} and $(c_{\mathrm{pc}}^i, K, \sigma_k, R, \sigma_r)$ via semantic security: $\mathbb{L}_R$: $(V_i^{id}, c_{\mathrm{pc}}^i, K, \sigma_k, R, \sigma_r)$ & $\mathbb{L}_R^{JCJ}$: (c_{pc}^i)

Hybrid 1. We replace Tally (Fig. 15, l. 27) with Ideal-Tally (Fig. 14, l. 17) by proving that C has no longer access to $\mathbb{L}_V$. We use a simulation-based approach to show that if an adversary with access to $\mathbb{L}_V$ has a non-negligible advantage over an adversary who does not have access to $\mathbb{L}_V$, then the decisional-Diffie-Hellman assumption is broken. The simulator gets as input a tuple of group elements that are either a Diffie-Hellman tuple or uniformly random and must output a guess. It interacts with C, simulates the honest parties of the protocol, and makes a guess based on its output.

The simulator then proceeds as follows:

1. **Setup:** The simulator receives its DDH challenge tuple (g, X, Y, Z) from an external challenger; they do not know the discrete logarithm of any component. It sets the system's public key as $\mathrm{pk} = (g, h = X)$ and broadcasts it.
2. **Coin Flip**: Flip the coin b.
3. **Adversarial Corruption**: The adversary chooses the set $\mathbf{V}_C$ of controlled voters and a target voter j. For each controlled voter and the target voter, the adversary chooses the number of fake credentials n_f to create. For the target voter, C sets the target vote β. If the number of controlled voters is not equal to n_C, or j is not an appropriate index then the simulator aborts.
4. **Registration**: For each voter i, the simulator runs the TRIP registration process while acting as the registrar. The simulator first issues each voter their real credential c_{sk}^i along with their public credential c_{pc}^i. The simulator then continues with generating fake credentials for each group of voters. For each controlled voter i in $\mathbf{V}_C$, the simulator issues n_f^i fake credentials as specified by the adversary. For the target voter, if $b = 0$, the simulator generates n_f^i+1 fake credentials for the target voter; otherwise, it generates n_f^i fake credentials. For each uncontrolled honest voter, the simulator creates fake credentials, for which the amount is sampled from a probability distribution that models the adversary's knowledge of the number of fake credentials that these voters intend to create. Finally, the simulator carries out check-out for all voters.
5. **Credential Release:** The simulator gives C the real and fake credentials of voters in $\mathbf{V}_C$. If $b = 0$, the simulator gives C the target voter's n_f^j+1 fake credentials; otherwise, the simulator gives C the voter's real credential and n_f^j fake credentials.
6. **Honest Ballot Casting**: For each honest voter i, the simulator samples a random vote $\beta_i \leftarrow\!\!\$\ D_{n_V - n_C, n_M}$ and posts a ballot for this vote on the voting ledger $\mathbb{L}_V$. The simulator forms the ballot using the input DDH tuple (g, X, Y, Z) as follows. To compute the encryption of the public credential c_{pc}^i, the simulator sets the ciphertext components as: $E_1 = (Y, Z \cdot c_{\mathrm{pc}}^i)$. To compute the encryption of the vote β_i, the simulator must use an independent set of random values. It can do this by creating a second, re-randomized challenge ciphertext, for instance by picking a random $s \in \mathbb{Z}_q$ and computing $E_2 = (Y \cdot g^s, Z \cdot X^s \cdot \beta_i)$. This way, if the input is Diffie-Hellman, the result is a valid ElGamal encryption, whereas if the input is random, the ciphertext is a random tuple and contains no information about the vote or the credential used to cast it. The simulator must now provide the accompanying NIZK proof that E_1 correctly encrypts a valid credential. Since the simulator does not know the randomness b (from $Y = g^b$) used to form E_1, it cannot generate the proof honestly. Instead, it leverages its control over the random oracle H:
   - For a given proof (e.g., a Schnorr-style proof of knowledge), the simulator chooses the final response z and challenge c first, picking them uniformly at random.
   - It then uses the public verification equation for the proof and works backwards to compute the commitment value A that would make the transcript (A, c, z) valid.
   - Finally, the simulator programs the random oracle, defining that for the input $(A, E_1, \mathit{statement})$, the output of H shall be c.

   The resulting proof is computationally indistinguishable from a real one for any adversary that can only query, but not control, the random oracle. The simulator follows the same procedure for all other required proofs and signatures.
7. **Adversarial Ballot Posting:** The adversary now posts a set of ballots onto $\mathbb{L}$.
8. **Decryption of Ballots:** The simulator can now check the NIZK proofs and discard the ballots with incorrect proofs. Then, since the simulator plays the role of the honest talliers, it can decrypt the ballots to prepare for the tallying process.
9. **Tallying Simulation:** This is carried out as in JCJ's simulator [77]. Namely, the simulator eliminates duplicates, mixes, removes fake votes and finally decrypts the remaining real votes:

- *Duplicate elimination:* The simulator removes duplicates.
- *Mixing:* To simulate the MixNets from the real tally protocol, the simulator outputs an equal-length list of random ciphertexts.
- *Credential Validity:* The simulator now checks, for each ballot, whether it was cast with a valid credential. This check is possible since the simulator can decrypt all the entries in $\mathbb{L}_R$ and $\mathbb{L}_V$.
- *Output final count:* The simulator can then use the decrypted values to compute the final tally and output it.

10. **Output Guess**: The simulator uses C's output to make its guess about whether the input was a DDH triplet or a random triplet.

If we can show that C has a non-negligible advantage in the real game over hybrid 1, then this implies that the simulator can break the DDH assumption. The key to this argument lies in how the simulator constructs the ballots in Step 6. When the input is a DDH tuple, where $X = g^a$, $Y = g^b$, and $Z = g^{ab}$ for unknown a, b, the public key given to C is $h = g^a$. The ciphertext E_1 is $(g^b, g^{ab} \cdot c^i_{\text{pc}})$. This is a mathematically valid ElGamal encryption of c^i_{pc} under the public key $h = g^a$, using b as the randomness. The same holds for E_2. Thus, when the input is DDH, the view of C corresponds to the actual game, and the adversary receives the contents of $\mathbb{L}_V$. However, if the input tuple is a random one, where $Z = g^c$ for a random c, then the ciphertext E_1 is $(g^b, g^c \cdot c^i_{\text{pc}})$. The second component, $g^c \cdot c^i_{\text{pc}}$, is a uniformly random group element from the adversary's perspective, as c is unknown and independent of a and b. In this scenario, the ciphertext perfectly conceals the credential, making the adversary's view equivalent to denying them access to the meaningful encrypted content of the ledger. Hence, if the adversary C holds a significant advantage in distinguishing the real game from the hybrid, then the probability that the simulator correctly guesses whether its input was DDH or random will also be significant, leading to a contradiction of the DDH assumption.

Hybrid 2. In this Hybrid, we demonstrate that the adversary's ability to demand a specific number of fake credentials from voters is lost, as compared to Hybrid 1. text In C-Resist (Fig. 15, l. 16 and 20), the voter gives their credentials to the adversary while in C-Resist-Ideal, the adversary always gets the voter's single real credential. (Fig. 14, l. 14).

First, we show that real and fake credentials are indistinguishable: both real and fake credentials contain a ZKP transcript, although the ZKP transcript for the fake credentials are simulated. The zero-knowledge property of the proof system implies indistinguishability.

Next, from hybrid 1, C does not get access to $\mathbb{L}_V$, thus C cannot use any real or fake credentials to cast valid ballots, or see ballots cast with these. As a result, since real and fake credentials are indistinguishable, always giving the adversary the real credential gives them no advantage since the credentials cannot influence the tally outcome with it. Now the value of b only determines whether or not the target voter casts a ballot.

For the same reason, C can only use the n_f fake credentials for determining whether the target voter cast a ballot or not and to influence the number of envelope challenges on $\mathbb{L}_E$.

Yet, the signing key pair $(\tilde{c}_{\text{sk}}, \tilde{c}_{\text{pk}})$ corresponding to each fake credential is sampled independently from the real pair $(c_{\text{sk}}, c_{\text{pk}})$, and these cannot be used by the adversary or the target voter to cast a valid vote. Regarding the number of challenges on $\mathbb{L}_E$, the only difference in the case where the target voter resists coercion is that they create an additional fake credential. As a result, the number of challenges will only differ by one. To detect this, the adversary needs to distinguish between the following distributions:

- Number of envelope challenges when target complies: $n_C + n_T + n_H + n_f + D^{fake}_{n_H}$
- Number of envelope challenges when target resists: $n_C + n_T + n_H + n_f + D^{fake}_{n_H} + 1$

Since everything but $D^{fake}_{n_H}$ is known to the adversary, this is equivalent to just distinguishing between $D^{fake}_{n_H}$ and $D^{fake}_{n_H} + 1$, so they get no advantage from requesting a certain number of fake credentials. Therefore, the adversary does not gain an advantage from specifying the number of fake credentials, since these fake credentials will not help them identify whether the voter gave them a real credential.

Hybrid 3. In this hybrid, we replace the TRIP roster $\mathbb{L}_R$ initialized on the first line of Figure 15 with the JCJ Roster $\mathbb{L}'_R$ from the Ideal Game in Figure 14.

To prove that the advantage of the adversary is negligible between these hybrids, we show that given a JCJ roster, a simulator can output a TRIP roster that is indistinguishable from a real one. This is possible due to the semantic security of ElGamal.

We describe the simulator: **Input:** JCJ roster $\mathbb{L}'_R$, List of voter IDs V_{id}

1. Create a kiosk key pair (k, K) and a registrar key pair (r, R). If there are multiple kiosks and registrars, these keys contain all the individual keys.
2. Initialize $\mathbb{L}'_R$ to contain each V_{id} in a different entry, along with a random timestamp d.
3. Apply a random permutation to the JCJ roster.
4. Append one entry V_e of the JCJ roster to each entry of $\mathbb{L}_R$.
5. For each entry, add the necessary signatures. First, use k to simulate the kiosk signature $\sigma_2 = \mathsf{Sig.Sign}_k(V^i_{id}||d||V_e)$ and append K, σ_{k_2} to the entry. Then, use r to simulate the registrar's check-out signature $\sigma_r = \mathsf{Sig.Sign}_r(V_{id}||d||V_e||\sigma_{k_2})$ and append R, σ_r to the entry.

6. Output $(K, R, \mathbb{L}'_R)$. Recall that we consider a single logical entity for all the registration actors.

As stated earlier, L'_R is indistinguishable from a real TRIP roster for the same list of voters due to the semantic security of ElGamal. If we start with a real TRIP roster, we could create intermediate rosters by swapping two encryptions at a time and updating the digital signatures until we get a random permutation. Each of these swaps will yield indistinguishable rosters by the semantic security of ElGamal. At the end, the distribution will be the same as that of our simulator.

Recall that the view of C is the list of valid verification keys for the kiosks and registrars, along with the TRIP roster. Since we have shown that the additional elements contained in the TRIP roster but not in the JCJ roster can be simulated, this means that the advantage of $\mathcal{A}$ is negligible.

With this, we have reached the Ideal JCJ game and have thus shown that the advantage of $\mathcal{A}$ in the C-Resist game is negligible over the advantage in C-Resist-Ideal.

□

L.2 Privacy

The privacy adversary seeks to uncover a voter's vote by identifying and decrypting the voter's real ballot. See our threat model for the capabilities of such an adversary (Appendix I.2). Unlike the coercion adversary, this adversary cannot directly interact with or influence voters; it must achieve its objective solely via electronic means. A voter's ballot is electronically accessible in three locations: on the voter's device, on the voting sub-ledger $\mathbb{L}_V$, and in the final tally. In what follows, we informally argue that the privacy adversary is unable to decrypt the voter's real ballot in these three locations.

Voter's device. Per our threat model (Appendix I.2), this adversary cannot compromise the device containing the voter's real credential. Compromising devices that hold fake credentials offers no advantage because ballots cast from these devices are not counted: the adversary cannot deduct the voter's real vote.

Voting sub-ledger $\mathbb{L}_V$. The voter's real ballot on the voting sub-ledger is encrypted, i.e., the has only access to the ciphertext of the real ballot. The adversary can identify the voter's real ballot on the ledger because it can compromise the registrar, including the kiosk that issued the target voter's credentials. Decrypting this ballot, however, would require the adversary to obtain the election authority's private key, which is impossible since the adversary can only compromise all but one authority member. Secret keys from compromised members provide no information into the collective key of the member that remains uncompromised, thus preventing reconstruction of the election authority's private key.

Final tally. In the final tally, ballots are revealed in decrypted form but through a mixing process that unlink ballots from voters. While the adversary can acquire the mixing permutation from all the compromised authority members, it cannot obtain it from the one member it cannot compromise. Each authority member generates and keeps their permutation secret, leaving no information for the adversary to deduce the permutation of the uncompromised member.

The privacy adversary is therefore unable to achieve its goal of revealing any voter's vote, thereby preserving privacy.

Excluded attack. As the Swiss Post e-voting system [8, Section 18.2], we exclude trivial attacks where all voters vote for the same option, which would indicate to the adversary what what the targeted voter voted. Also, we do not consider the adversarial strategy to delete from the voting sub-ledger and final tally all but one vote, to learn the targeted ballot from the tally's result. We assume this attack to be detectable and thus inapplicable by the privacy adversary.

L.3 Individual Verifiability

In this section, we provide a formal security analysis of the individual verifiability (IV) property for TRIP. Informally, IV ensures that an honest voter can detect any adversarial attempt to tamper with either (1) the credential produced during registration or (2) the ballot that will eventually be tallied. Appendix L.3.1 revisits the threat model tailored to individual verifiability. Appendix L.3.2 formalizes the IV security game. Appendix L.3.3 maps TRIP's procedures to the formal model. Appendix L.3.4 states the main security theorem followed by its proof. Appendix L.3.5 presents the implications of the adversary's advantage. Appendix L.3.6 presents our new security definitions, iterative individual verifiability, and strong iterative individual verifiability, which take into account the adversary's explicit goal of altering the election outcome. We also show that TRIP satisfies strong iterative individual verifiability.

L.3.1 Threat Model The adversary's objective is to cause the final registration or ballot associated with an honest voter on the ledger to be inconsistent with that voter's actions, without this inconsistency being detected by the voter or their VSD.

The honest parties are the voter and their personal device, VSD, which follow the protocol exactly. This includes the voter following the procedural steps required during in-person registration, such as completing the correct interactive sequence of the Σ-protocol.

The integrity adversary $\mathcal{I}$ is computationally bounded and controls every actor other than the voter and their VSD. This includes other voters, registration kiosks, printers, officials, the election authority and the public ledger. In essence, $\mathcal{I}$ has complete, active control over the entire election infrastructure, and consequently possesses all long-term secret keys. The adversary, however, cannot interfere with the physically printed envelopes once the voter has entered the booth for their registration session without being detected.

The public ledger $\mathbb{L}$ is modeled as an ideal, append-only functionality accessible to all parties. The adversary $\mathcal{I}$ may append arbitrary entries but cannot delete, modify or reorder existing ones.

L.3.2 Formal Definition. We now present our formal definition, building on prior work such as Swiss Post E-Voting System [8], and CH-Vote [29]. We first define the syntax of a generic voting scheme Π — an election scheme without tallying — and then use it to construct our security game, IV.

A voting scheme Π is a tuple of four algorithms:

- $\mathsf{Setup}_{\mathcal{I}}(\mathsf{params}) \rightarrow (\mathsf{pp}, \mathbb{L}, \mathbb{V}, \mathsf{st}_{\mathcal{I}})$: An algorithm run by the adversary $\mathcal{I}$ that takes as input the game's public parameters params which are assumed to be honestly generated and hardcoded into the game. It outputs scheme-specific public parameters pp, the initial ledger state $\mathbb{L}$, the list of the eligible voters $\mathbb{V}$ and the adversary's state $\mathsf{st}_{\mathcal{I}}$ containing any secret keys (like the authority's secret key $\mathbb{A}_{\mathsf{sk}}$ to determine the game's outcome).
- $\mathsf{VoterInteraction}(V \leftrightarrow \mathit{infra})$: An interactive protocol between the honest voter's agency (represented by an oracle, V) and the malicious infrastructure *infra* controlled by the adversary $\mathcal{I}$. This protocol encapsulates registration and voting, resulting in a voter receipt π, and a possibly modified ledger $\mathbb{L}$.
- $\mathsf{VoterVerify}(\mathsf{params}, \mathsf{pp}, \mathbb{L}, \pi) \rightarrow \{\top, \bot\}$: A deterministic algorithm executed on VSD. It takes as input the public parameters params, the scheme-specific parameters pp, the final public ledger $\mathbb{L}$ to be used for tallying, and the voter's receipt π. It outputs $\top$ (accept) if the voter's interactions are correctly reflected on the ledger, and $\bot$ (reject) otherwise.
- $\mathsf{Extract}(\mathsf{params}, \mathsf{pp}, \mathbb{L}, \mathsf{st}_{\mathcal{I}}, V_{id}) \rightarrow \mathsf{goal}$: A deterministic algorithm used only inside the security game that takes as input the public parameters params, scheme-specific parameters pp, the final ledger $\mathbb{L}$, the adversary's state $\mathsf{st}_{\mathcal{I}}$ and a voter identifier V_{id}. It outputs the recorded information associated with that voter V_{id} on the ledger or $\bot$ if no valid record exists.

We formalize individual verifiability using the security game Game IV, shown in Fig. 16. The game begins with the adversary $\mathcal{I}$ running Setup. Following this, $\mathcal{I}$ chooses a target voter $V^{\star}_{id}$, an honest intent for that voter, and a malicious goal it wishes to have recorded on the ledger, where goal $\neq$ intent.

The core of the game is the interaction phase, where $\mathcal{I}$ is given oracle access to $V(V^{\star}_{id}, \mathsf{intent})$ which perfectly simulates the honest voter's actions. At the end of the interaction, $\mathcal{I}$ outputs the voter's receipt π and the final ledger $\mathbb{L}$.

The adversary wins if its malicious goal is recorded on the ledger (Extract returns goal) *and* the voter's local verification on VSD passes (VoterVerify returns $\top$). We define the advantage of an adversary $\mathcal{I}$ against individual verifiability of scheme Π as:

$$Adv^{\mathsf{IV}}_{\Pi,\mathcal{I}} = \Pr[\mathsf{Game\ IV}_{\Pi,\mathcal{I}} = 1]$$

```
Game $\mathsf{IV}_{\Pi,\mathcal{I}}(\mathsf{params})$
1 : $(\mathsf{pp}, \mathbb{L}, \mathbb{V}, \mathsf{st}_{\mathcal{I}}) \leftarrow \Pi.\mathsf{Setup}_{\mathcal{I}}(\mathsf{params})$
2 : $(V^{\star}_{\mathsf{id}}, \mathsf{intent}, \mathsf{goal}) \leftarrow \mathcal{I}(\text{Choose Target \& Goal})$
3 : $(\pi, \mathbb{L}) \leftarrow \mathcal{I}^{V(V^{\star}_{\mathsf{id}}, \mathsf{intent})}(\mathbb{L})$
4 : $\mathsf{happy} \leftarrow \Pi.\mathsf{VoterVerify}(\mathsf{params}, \mathsf{pp}, \mathbb{L}, \pi)$
5 : $\mathsf{adv\_outcome} \leftarrow (\mathsf{Extract}(\mathsf{params}, \mathsf{pp}, \mathbb{L}, \mathsf{st}_{\mathcal{I}}, V^{\star}_{\mathsf{id}}) = \mathsf{goal})$
6 : return 1 if $(\mathsf{happy} \wedge \mathsf{adv\_outcome} \wedge (\mathsf{goal} \neq \mathsf{intent}))$ else 0
```

Figure 16. Individual Verifiability (Ind-Ver). An adversary wins if it can make the ledger reflect its malicious goal for an honest voter, without the voter's verification procedures detecting the manipulation.

L.3.3 Protocol Instantiation. We now map the abstract syntax to the concrete procedures of Π_{TRIP}.

The global public parameters are params $= (G, q, g,\ H)$ (G, q, g) is a cyclic group of prime order q where the DDH problem is hard, and $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is a hash function. These are assumed to be honestly set to prevent the adversary from winning vacuously.

Setup is executed by $\mathcal{I}$, taking params as input. $\mathcal{I}$ may generate arbitrary keys for the election authority, officials, kiosks, and printers, and may populate the ledger $\mathbb{L}$ and voter list $\mathbb{V}$ in any way. It outputs the scheme-specific parameters pp which would include the public keys of all parties.

The VoterInteraction abstract protocol is instantiated by two concrete interactions: the in-person TRIP registration procedure (Fig. 6, lines 1-9), and the subsequent "at-home" Vote procedure (Fig. 12). The adversary $\mathcal{I}$ controls the infrastructure side of both interactions. For the registration procedure, the voter — not their VSD— performs the interactive choices, such as selecting the number of credentials n_c (line 4), and the final credential for check-out c_v (line 8). The pick of n_c envelopes is a sequential process, where the voter picks one envelope for RealCred and then $n_c - 1$ envelopes for FakeCred. The voter aborts the game, returning 0, if the adversary deviates from the Σ-protocol during RealCred. For the voting procedure, VSD also keeps the computed ballot B in memory for the VoterVerify procedure.

The voter receipt π corresponds to the full set of data contained in the n_c physical paper credentials (QR codes) the voter takes home after registration.

The abstract $\mathsf{VoterVerify}(\mathsf{params}, \mathsf{pp}, \mathbb{L}, \pi)$ protocol is instantiated as a procedure run on the voter's trusted VSD, taking the final public ledger $\mathbb{L}$ to be tallied and the registration receipt π as input. It returns $\top$ if and only if both of the following checks succeed:

- **Registration Verification**: VSD runs the Activate procedure for every credential contained in the receipt π (Fig. 6, lines 10-11). This includes verifying the correctness of the zero-knowledge proof constructed during RealCred, and the first-time redemption of the envelope's nonce against $\mathbb{L}_E$. This check succeeds only if this loop completes for all credentials created without aborting.
- **Vote Verification**: VSD retrieves the locally stored ballot B_{local} that it cast during the Vote procedure. It queries the ledger $\mathbb{L}$ for the ballot associated with the voter's credential c_{pk}. Let this be B_{posted}. This check succeeds only if the ballot posted on the ledger B_{posted} is bit-for-bit identical to B_{local}. If multiple ballots are posted for c_{pk} it retrieves the one that will be tallied — based on some policy, which for this instantiation, we set as the last ballot cast counts.

The game's Extract algorithm determines the final outcome for a voter as recorded on the ledger. It first parses the registration ledger $\mathbb{L}_R[V_{id}]$ to find the public credential c_{pc}. It then decrypts c_{pc} using the authority key $\mathbb{A}_{\text{sk}}$ from $\text{st}_{\mathcal{I}}$ to recover the associated credential public key, c'_{pk}. It continues by finding the ballot that counts (i.e., last ballot cast) B_{posted} on the voting ledger $\mathbb{L}_V$ associated with c'_{pk}. Finally, it decrypts B_{posted} using $\mathbb{A}_{\text{sk}}$ to recover the vote message m'. If any of the steps fail, it returns $\perp$. Otherwise, it outputs the tuple (c'_{pk}, m'). The intent and goal are thus defined as tuples of the voter's credential public key and the choice of vote.

L.3.4 Security Theorem. We now state our main theorem regarding the individual verifiability of our protocol.

Theorem 3. *Let Π_{TRIP} be instantiated as described in Appendix L.3.3. If the underlying Σ-protocol is computationally sound (which relies on the hardness of the Discrete Logarithm Problem), and the ElGamal scheme is perfectly binding, then, for any PPT adversary $\mathcal{I}$ playing Game* IV*, its advantage $Adv^{\mathsf{IV}}_{\Pi_{TRIP},\mathcal{I}}(\lambda)$ is bounded by:*

$$Adv^{\mathsf{IV}}_{\Pi_{TRIP},\mathcal{I}}(\lambda) \leq \max_{1\leq k\leq n_E} E_{n_c\sim D_C}\left[\frac{k}{n_E}\cdot\frac{\binom{n_E-k}{n_c-1}}{\binom{n_E-1}{n_c-1}}\right] + \mathsf{negl}(\lambda)$$

where n_E is the total number of physical envelopes present in the booth during the voter's registration, k is the number of those envelopes maliciously prepared by $\mathcal{I}$, n_c is the number of credentials the voter chose to create, and D_C is the distribution of the voter's choice for n_c.

Proof. Let Win be the event that $\mathcal{I}$ wins the Game IV. In other words, Win means that VoterVerify returns $\top$ and Extract returns the adversary's goal. The adversary can win by tampering with registration or by tampering with the vote. Let RegSub be the event that the adversary's goal for the credential key, goal.c_{pk}, differs from the honest intent, intent.c_{pk}: goal.$c_{\text{pk}} \neq$ intent.c_{pk}. Let VoteSub be the event that goal.$m \neq$ intent.m where m represents the vote.

For the adversary to win, the goal must differ from intent, which means that either RegSub or VoteSub (or even both) must occur. Therefore, the Win event is a sub-event of (RegSub$\vee$ VoteSub), allowing us to partition the Win event into two disjoint cases based on whether registration was tampered. By the law of total probability, we get:

$$\begin{aligned}\Pr[\mathsf{Win}] &= \Pr[\mathsf{Win} \wedge (\mathsf{RegSub} \vee \mathsf{VoteSub})]\\ &= \Pr[\mathsf{Win} \wedge \mathsf{RegSub}]\\ &\quad + \Pr[\mathsf{Win} \wedge \neg\mathsf{RegSub} \wedge \mathsf{VoteSub}]\end{aligned}$$

We bound each term in a separate lemma, Lemma 4 and Lemma 5, respectively.

Lemma 4. *The probability of an adversary $\mathcal{I}$ winning Game* IV *via registration tampering is bounded by:*

$$\Pr[\mathsf{Win} \wedge \mathsf{RegSub}] \leq \max_{1\leq k\leq n_E} E_{n_c\sim D_C}\left[\frac{k}{n_E}\cdot\frac{\binom{n_E-k}{n_c-1}}{\binom{n_E-1}{n_c-1}}\right] + \mathsf{negl}(\lambda)$$

Proof. The proof proceeds via a game hop argument. Let Win_i be the event that the adversary wins Hybrid i.

Hybrid 0. This is the real Game IV. The adversary $\mathcal{I}$ interacts with the game as defined in the TRIP protocol. The adversary's success probability is $\Pr[\mathsf{Win}_0 \wedge \mathsf{RegSub}]$.

Hybrid 1. This hybrid is identical to Hybrid 0 except that it aborts if the adversary $\mathcal{I}$ produces two different plaintexts (e.g., credential public keys) that are valid openings to the same ElGamal ciphertext c_{pc}. Since ElGamal is perfectly binding, this event is information-theoretically impossible. Thus,

$$|\Pr[\mathsf{Win}_0 \wedge \mathsf{RegSub}] - \Pr[\mathsf{Win}_1 \wedge \mathsf{RegSub}]| = 0$$

Hybrid 2. This hybrid is identical to Hybrid 1 except we replace the Σ-protocol with an idealized proof system that aborts if $\mathcal{I}$ tries to produce a proof for a false statement. The statement being proven in RealCred is knowledge of the randomness x used to create c_{pc} as the encryption of c_{pk}.

$$\mathsf{ZKPoE}_{C_1,X}\{(x) : C_1 = g^x \wedge X = \mathbb{A}_{\text{pk}}^x\}$$

where $c_{\text{pc}} = (C_1, C_2)$ and $c_{\text{pk}} = \frac{C_2}{X}$. An adversary attempting to tamper with registration wants to get on the ledger c_{pc} that encrypts some other c'_{pk} — known only to $\mathcal{I}$ — instead of the voter's intended public key c_{pk}. This is a false statement, so the probability that the adversary wins in Hybrid 2 is

$$\Pr[\mathsf{Win}_2 \wedge \mathsf{RegSub}] = 0$$

. We now wish to bound the advantage of $\mathcal{I}$ between Hybrid 1 and Hybrid 2. To do this, we must adapt the soundness analysis of the Schnorr Σ-protocol to our setting where adversarially chosen envelopes are used for the challenge instead of a uniformly random group element. The adversary can break soundness by either guessing the envelope challenge or finding the corresponding discrete log that allows them to produce the desired ciphertext. This second case is oblivious to how the challenge is chosen, and puts us back in the standard computational soundness analysis of the Schnorr Σ-protocol. By reduction to the hardness of the Discrete Logarithm Problem, the probability of generating a valid proof in this case is negligible.

Now we must bound the probability that $\mathcal{I}$ is able to guess the challenge e that will be selected by the voter. Recall that our threat model allows $\mathcal{I}$ to adversarially select the set of envelopes available to the voter. We first show that the best strategy that $\mathcal{I}$ can adopt is to have a malicious set of credentials $\mathcal{M}$ containing the same repeated nonce, and an honest set $\mathcal{H}$ consisting of envelopes with a unique nonce.

Consider an arbitrary strategy where the adversary prepares n_E envelopes consisting of a malicious set $\mathcal{M}$ and an honest set $\mathcal{H}$. We partition $\mathcal{M}$ into subsets $\mathcal{M}_1, \cdots \mathcal{M}_k$ where $\mathcal{M}_i$ contains various envelopes with the same nonce e_i^*.. Now, when guessing the nonce the voter will choose, let p_i be the probability that the adversary guesses e_i^*. Then, let $P = \sum p_i$, so the probaility that the adversary tries to guess an envelope from $\mathcal{H}$ is $1 - P$. The adversary wins if it correctly guesses the envelope picked by the voter for their real credential *and* the voter only picks envelopes with unique nonces. We show that this probability is maximal when all the envelopes in $\mathcal{M}$ have the same repeated nonce.

Now assume the voter creates n_c total credentials. We can upper bound the probability that the adversary correctly guesses the voter's real credential without getting caught as follows. Let EnvGuess be the event where the adversary correctly guesses the nonce and let DupCatch be the event where the voter finds duplicate nonces. First, we lower bound the probability that the adversary gets caught. Note that for all $\mathcal{M}_i$, this probability is at least the probability that it gets caught on a specific $\mathcal{M}_i$. This tells us that

$$\Pr[\mathsf{DupCatch} \mid \mathsf{EnvGuess}] \geq \max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}]$$

which gives us that

$$\Pr[\neg\mathsf{DupCatch} \mid \mathsf{EnvGuess}] \leq 1 - \max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}]$$

Then, the probability of success is

$$\Pr[\neg\mathsf{DupCatch} \wedge \mathsf{EnvGuess}]$$
$$\leq (1-\max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}])\cdot\Pr[\mathsf{EnvGuess}]$$

By Union Bound:

$$\leq (1-\max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}])\cdot\left(\sum_{i=1}^{k} p_i \frac{|\mathcal{M}_i|}{n_E}\right)$$

$$\leq (1-\max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}])\cdot\left(\sum_{i=1}^{k} \frac{|\mathcal{M}_i|}{n_E}\right)$$

$$\leq (1-\max_i \Pr[\mathcal{I} \text{ caught on } \mathcal{M}_i \mid \mathsf{EnvGuess}])\cdot\frac{|\mathcal{M}|}{n_E}.$$

Now, note that if $\mathcal{M}$ consists of envelopes with a single repeated nonce, then this actually becomes an equality. We can therefore conclude that an adversary gains nothing from using various repeated nonces.

Now, we analyze the optimal strategy, which is:

1. $\mathcal{I}$ prepares n_E envelopes composed of two sets: a malicious set $\mathcal{M}$ of k envelopes containing the same, pre-determined nonce $e^\star$, and an honest set $\mathcal{H}$ of $n_E - k$ envelopes containing unique nonces.
2. During the RealCred interaction, $\mathcal{I}$ must commit to Y_c *before* seeing the voter's challenge e. Therefore, to prove a false statement that c_{pc} encrypts intent.c_{pk} when it actually encrypts goal.c_{pk}, $\mathcal{I}$ must guess the challenge e that the voter will choose. Thus, $\mathcal{I}$ hopes that the voter chooses an envelope from $\mathcal{M}$ (containing $e^\star$) for this step.
3. The VoterVerify procedure checks for reused nonces across all envelopes posted on the ledger $\mathbb{L}_E$. Therefore, to pass this check, $\mathcal{I}$ must hope that the voter during FakeCred picks only envelopes from the honest set $\mathcal{H}$.

The adversary wins if the voter chooses an envelope from $\mathcal{M}$ for RealCred — probability of $\frac{k}{n_E}$ *and* chooses the subsequent $n_c - 1$ envelopes for FakeCred entirely from $\mathcal{H}$. Given the first choice was from $\mathcal{M}$, there are $n_E - 1$ envelopes left, of which $n_E - k$ are in $\mathcal{H}$. The probability of the second event is then:

$$\frac{\binom{n_E-k}{n_c-1}}{\binom{n_E-1}{n_c-1}}$$

Given that these are two distinct and ordered probabilistic events, the total advantage of the adversary in Hybrid 1 over Hybrid 2 (and thus its probability of success) is, maximizing over $\mathcal{I}$'s choice of k and averaging over the voter's choice of n_c, is:

$$\Pr[\mathsf{Win}_1 \wedge \mathsf{RegSub}] = \max_{1\leq k\leq n_E} E_{n_c\sim D_C}\left[\frac{k}{n_E}\cdot\frac{\binom{n_E-k}{n_c-1}}{\binom{n_E-1}{n_c-1}}\right] + \mathsf{negl}$$

This probability represents the adversary's optimal strategy in a game of uncertainty. The adversary must choose the number of malicious envelopes, k, without knowing how many credentials, n_c, the voter will choose to create. A greedy strategy of setting $k = n_c$ (all envelopes being malicious) would guarantee a win if the adversary knew that $n_c = 1$, as the nonce-reuse check in VoterVerify would never detect duplicates assuming the adversary issues new envelopes after each registration session. However, this strategy fails spectacularly if $n_c > 1$, as the VSD would immediately detect a nonce-reuse on the activation of the second credential. Since $\mathcal{I}$ does not know n_c, it must balance the potential reward of subverting the RealCred interaction against the risk of detection during FakeCred procedures. This formula captures the maximum expected success for $\mathcal{I}$, averaged over the voter's behavioral distribution D_C.

Summing this statistical term with the negligible terms from the game hops conclude the proof of this lemma. $\square$

Lemma 5. *Given that registration was not subverted (*$\neg\mathsf{RegSub}$*), the probability of an adversary winning via vote manipulation is* 0. $\Pr[\mathit{Win} \land \neg\mathsf{RegSub} \land \mathsf{VoteSub}] = 0$

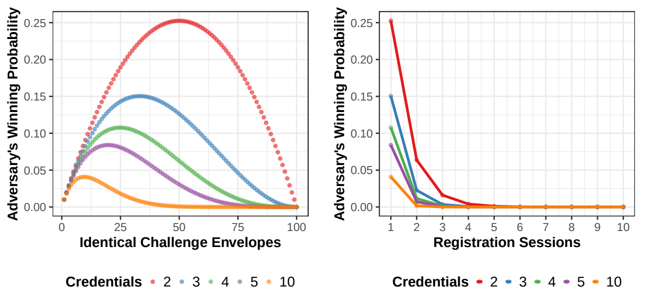
Proof. We proceed by direct argument. The condition $\neg\mathsf{RegSub}$ implies that the public credential c_{pc} on the ledger correctly encrypts the honest credential key c_{pk} that the voter possesses. The adversary $\mathcal{I}$, however, knows the credential's secret key as it generated the key pair $(c_{\text{sk}}, c_{\text{pk}})$ on the kiosk during RealCred. The adversary's goal is to have the final tallied vote be goal.m while the voter's intent was intent.m, without the VoterVerify check failing. We consider all possible strategies for $\mathcal{I}$.

The first strategy is to submit a ballot on the ledger given $\mathcal{I}$ knowledge of c_{sk}. $\mathcal{I}$ constructs and signs a ballot $B_{\mathcal{I}}$ for the vote goal.m. $\mathcal{I}$ then sends this ballot to ledger $\mathbb{L}_V$ such that it will be the one extracted for the tally. The game requires $\mathcal{I}$ to output the final ledger $\mathbb{L}$ *before* VoterVerify is run, as a means of mimicking continuous checking by VSD. If $\mathcal{I}$ posts only its ballot $B_{\mathcal{I}}$, the voter's VSD will detect a ballot B associated with c_{pk} on L_V when no vote was cast and fail. If the voter's ballot is posted B_V, and then $\mathcal{I}$ posts $B_{\mathcal{I}}$, VoterVerify will compare the last ballot $B_{\mathcal{I}}$ with its locally stored ballot B_V. Since $B_V \neq B_{\mathcal{I}}$, the check will fail. If $\mathcal{I}$ posts $B_{\mathcal{I}}$ and then the voter's ballot is posted B_V, the last ballot will be B_V, and the Extract function will find intent.m, not the adversary's goal.m. Therefore, $\mathcal{I}$ does not win in all cases.

The other strategy is to modify the decryption of the ballot. $\mathcal{I}$ intercepts or appends a new ballot based on the voter's honestly generated ballot B_V (containing an encryption of intent.m) and attempts to modify it into $B_{\mathcal{I}}$ such that $B_{\mathcal{I}}$ decrypts to goal.m. For VoterVerify to pass, $B_{\mathcal{I}}$ must be bit-for-bit identical to the original B_V that the VSD stored locally. However, since the ElGamal encryption is perfectly binding, $\mathcal{I}$ cannot get the ciphertext to decrypt to anything but the voter's plaintext intent.m. It is information-theoretically impossible to find a different message goal.m that is a valid opening of the same ciphertext. Therefore, any modification that changes the underlying vote will necessarily change the ciphertext, causing the bit-for-bit check in VoterVerify to fail. The probability of success for this strategy is 0.

In all scenarios, the combination of the VSD's bit-for-bit check and the perfect binding property of the encryption scheme ensures that any attempt by $\mathcal{I}$ to change the vote after a successful registration will be detected. The probability of success is therefore 0. $\square$

We now combine the results of Lemma 4 and Lemma 5 to complete the proof. The total advantage of the adversary $\mathcal{I}$ is the sum of its advantages in the two disjoint scenarios we analyzed:



(a) Adversary's Winning Probability based on the k identical envelopes & n_c voter-chosen credentials **(b)** Adversary's Winning Probability across Voter Registration Sessions & Number of Credentials Voter Creates.

Figure 17. Adversary's Winning Probability for 100 Envelopes.

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IV}}_{\Pi_{TRIP},\mathcal{I}} &= \Pr[\mathsf{Win}] \\ &= \Pr[\mathsf{Win} \land \mathsf{RegSub}] \\ &\qquad + \Pr[\mathsf{Win} \land \neg\mathsf{RegSub} \land \mathsf{VoteSub}] \\ &\leq \left(\max_{1\leq k\leq n_E} \mathbb{E}_{n_c\sim D_C}\left[\frac{k}{n_E} \cdot \frac{\binom{n_E-k}{n_c-1}}{\binom{n_E-1}{n_c-1}} \right] + \mathsf{negl}(\lambda) \right)\end{aligned}$$

This concludes the proof of the theorem. $\square$

L.3.5 Implications of the Security Bound Beyond the proof, we can build intuition for the adversary's security bound by analyzing their strategic options. To subvert a voter's registration without being detected cryptographically, the adversary must forge a zero-knowledge proof during the RealCred procedure. This requires knowing the voter's ZKP challenge *before* committing to the proof, as is done for FakeCred. The adversary's only viable strategy to achieve this is to "poison" the deck of envelopes in such a way that they can predict which challenge the voter will select.

The Adversary's Dilemma The core of the adversary's strategy is to place multiple envelopes containing the same, pre-selected nonce $e^\star$ in the booth: the malicious set $\mathcal{M}$. Their hope is that the voter, when creating their real credential, will happen to pick up one of these malicious envelopes. This strategy, however, presents a fundamental dilemma, a trade-off between maximizing the chance of success and minimizing the risk of immediate detection. This is precisely the dynamic captured by the security bound in Theorem 3 and visualized in Figure 17.

Consider the adversary's choice of how many malicious envelopes, k, to place in the booth our of a total of n_E. As an adversary increases k, the probability that the voter selects a malicious envelope for their real credential, $\Pr[e^\star \in \mathcal{M}]$, increases linearly k/n_E. This is the adversary's path to a successful tampering. The voter's primary defense is the creation of fake credentials — not just individual verifiability, but also for coercion-resistance! For each fake credential, the voter picks another envelope from the remaining pool and if *any* of these additional draws also comes from the malicious set, then VSD will detect a reused nonce during activation and the attack will fail. As k increases, so does the probability of VSD's detection.

This tension is illustrated in Figure 17a. For any given number of credentials the voter creates, the adversary's winning probability initially rises with k, but then decreases as the risk of detection increases. The peak of each curve represents the adversary's optimal choice of k, but the adversary does not know which curve they are on when selecting k as it depends on the voter's choice of n_c. The expectation over the voter's choice distribution D_X captures this uncertainty and significantly limits the adversary's real-world advantage. The user study (Appendix P) demonstrates the soundness of this notion: 76% of participants chose to create at least one fake credential, with 53% stating they would create fake credentials if such a system existed today.

Large-Scale Fraud The adversary's difficulty in winning compounds dramatically when attempting to influence an entire election. To succeed, they must win the registration gamble not just for one targeted voter, but for *every* single voter they target, as a single detected failure could expose the entire operation. If the maximum probability of deceiving a single voter is $p_{\max}$, the probability of deceiving N independent voters is $p_{\max}^N$.

This exponential decay is visualized in Fig. 17b. Even with an optimistic success probability of $\sim 25\%$ for a single voter (the peak for $n_c = 2$), the chance of successfully subverting just three such voters undetectably drops to less than 2%. This shows that the success probability makes large-scale fraud infeasible against a population of even moderately cautious voters. This fact, where security against one is amplified into security against many, is what motivates the concept of *iterative individual verifiability* (Appendix L.3.6).

Additional Adversarial Behaviors An adversary might exploit human biases — humans being non-random — by, for example, placing their malicious envelopes at the top of the stack. However, even if this could double or triple the success rate against one voter, it provides only a constant-factor advantage against an exponential problem when targeting many voters.

The adversary could also program the kiosk to abort if the voter selects an "un-guessed" envelope. This, however, is not an undetectable subversion but rather an observable and undeniable failure. A single voter reporting such behavior is sufficient to prove misconduct. The authorities are capable of challenging the kiosk to reveal the ZKP commitment secret y for the Y_c it printed in the first QR code (Fig. 9a, line 5) — a computationally infeasible task if it cheated. If the kiosk refuses to reveal y or provides an incorrect value, authorities gain unequivocal proof of misconduct.

```
Game I-IV$^{\lambda_v}_{\Pi,\mathcal{I}}$(params)
1 : $\mathbb{V} \leftarrow \mathcal{I}$(% Compile electoral roll)
2 : successes $\leftarrow 0$
3 : for $V_{id}$ in $\mathbb{V}$ do
4 :     successes += Game IV$^*_{\Pi,\mathcal{I}}$(params, $V_{id}$)
5 : endfor
6 : return 1 if (successes $\geq \lambda_v$) else 0
```

Figure 18. Iterative Individual Verifiability (I-IV) Definition of iterative individual verifiability to account for an adversary's goal of altering an election outcome.

Ultimately, while the adversary has a non-negligible advantage of winning against a single voter, this advantage becomes negligible when attempting to scale the attack.

L.3.6 Iterative Individual Verifiability

While the IV game establishes the security for a single voter, the adversary's true objective is not to deceive one voter but to deceive enough voters to alter an election's final outcome. This requires a model of security that accounts for an adversarial campaign across a population of voters. To change an election outcome, they must successfully and undetectably subvert a number of votes λ_v equal to or greater than the election's margin of victory.

We explore this requirement through two complementary views. First, we discuss a general model applicable to any voting protocol, resulting in a standard binomial security bound. Second, we leverage a specific property of TRIP— the ability to produce non-repudiable proof of misbehavior — to define a "high-stakes" game, S-I-IV, which yields a much stronger exponential security bound for TRIP.

Formal Definition: Iterative Individual Verifiability. We define the I-IV property using the security game Game I-IV shown in Fig. 18. This game is a wrapper that repeatedly calls a voter-targeted version of our base IV game, which we now formally define.

The I-IV game requires a sub-routine that forces the adversary to attack a specific, pre-determined voter. We denote this game as Game IV*. It is identical in all respects to Game IV from Figure 16 but with two modifications: First, the game now takes as an additional input parameter V_{id} and hardcodes the target voter as $V^\star_{id} \leftarrow V_{id}$. Then, the step $(V^\star_{id}, \text{intent}, \text{goal}) \leftarrow \mathcal{I}(\text{Choose Target \& Goal})$ becomes $(\text{intent}, \text{goal}) \leftarrow \mathcal{I}(\text{Choose Goal})$.

With this sub-game defined, we now present the main game in Fig. 18. In this game, the adversary first commits to an electoral roll $\mathbb{V}$. The challenger then executes $|\mathbb{V}|$ independent rounds of IV*, one for each voter $V_{id} \in \mathbb{V}$. The adversary wins if they accumulate at least λ_v successful attacks.

General Model for Iterative Attacks For any voting scheme where a detected failure might be deniably attributed to a "systems fault", an adversary's campaign can be modeled as a sequence of Bernoulli trials. An adversary might therefore attack $n_V > \lambda_v$ voters, tolerating some failed attempts to maximize their chance of reaching λ_v wins. Each attempt against a single voter is a trial with maximum success probability $p_{\max}$. The adversary's success in this scenario is bounded by the tail property of the binomial distribution $B(n_V, p_{\max})$:

$$\Pr[\text{successes} \geq \lambda_v] \leq \sum_{i=\lambda_v}^{n_V} \binom{n_V}{i} (p_{\max})^i (1 - p_{\max})^{n_V - i}$$

This bound reflects a weaker security guarantee, as an adversary can trade a higher number of attempts for a greater probability of success.

Strong Iterative Individual Verifiability TRIP provides a much stronger guarantee that allows us to move beyond the general binomial model. Specifically, any detected attempt at tampering provides non-repudiable proof of the adversary's malicious actions. We now show this in the following lemma.

Lemma 6. *Let $\perp$ be the event that* VoterVerify *returns reject for an honest voter's interaction in TRIP. This event constitutes non-repudiable cryptographic proof of misbehavior, which an adversary can only deny culpability for this event by breaking the hardness of the Discrete Logarithm Problem or the EUF-CMA security of the underlying signature scheme.*

Lemma 6 Proof Sketch. A rejection by VoterVerify($\perp$) during the registration phase of TRIP arises from a failed attempt to deceive the voter. We can disregard failures during the voting phase, as Lemma 5 of Theorem 3 shows that the adversary's success probability is 0 in that phase. The registration failures can be categorized into two main types: ZKP soundness failure and nonce-reuse detection.

As established in Theorem 3, to cheat the RealCred ZKP, the adversary must commit to Y_c before seeing the voter's challenge e. This forces the kiosk to guess a challenge $e^\star$. If the voter selects an envelope with a different challenge $e \neq e^\star$, the adversary has two options:

First, it may abort the protocol. The voter is left with an incomplete but signed receipt (i.e., the first QR containing the commitment Y_c). Upon reporting the voter reporting the kiosk's malfunction, authorities can challenge the kiosk to reveal the ZKP secret y corresponding to Y_c. A malicious kiosk, having computed Y_c to cheat, will be unable to produce a valid y that satisfies the ZKP commitment equation: $Y_c = (Y_1, Y_2) = (g^y, \mathbb{A}_{\mathsf{pk}}^{y})$, which is a failure reducible to the hardness of the DLP.

The adversary may also complete the protocol with an invalid proof. If the adversary provides an invalid ZKP response r on the receipt, the voter's VSD will detect this during the Activate procedure, triggering the $\perp$ event. The voter now possesses, at the minimum, a receipt containing a signed ZKP commitment in the first QR code and an invalid ZKP response r in the third QR code. This is also non-repudiable proof that the kiosk acted maliciously unless the adversary can shift the blame to a third party by demonstrating that they can break the EUF-CMA security of the signature scheme to claim forgery.

The second class of attacks is the nonce-reuse detection. Suppose the adversary successfully guesses the challenge for RealCred (i.e., the voter picks an envelope with $e^\star$). They must still pass the $n_c - 1$ FakeCred checks. If the voter selects another malicious envelope containing $e^\star$, the kiosk similarly has two options: abort or complete the procedure.

If the kiosk aborts during FakeCred, the authorities check that each envelope used has a unique challenge. Furthermore, for additional verification, the authorities can perform the same ZKP soundness check as described earlier using the voter's stated real credential. An honest kiosk can always comply with the challenges faced by the authorities by retaining the secrets produced during their interaction with the voter until the end of that interaction.

If the kiosk completes the protocol, the VSD will later detect a nonce re-use during the Activate procedure, triggering a $\perp$. This detection is non-repudiable because each physical envelope is supposed to contain a unique challenge, signed by an (adversary-controlled) envelope printer. The voter can now present these distinct envelopes with the same challenge to the authorities, providing a direct proof of malicious behavior.[11]

In all scenarios, a $\perp$ event provides the voter and the authorities with cryptographic evidence of misbehavior. Therefore, the adversary's failure to tamper is non-repudiable, bounded only by the negligible probability of breaking DLP and the EUF-CMA property of the underlying signature scheme. □

This Lemma 6 allows us to model a game where the entire campaign fails upon a single detection. Furthermore, it implies that a rational adversary has no incentive to continue attacking once their goal of λ_v wins is met, as any further attempts only add risk for any $p_{\max} < 1$. We formalize this with the game S-I-IV.

Formal Definition: Strong Iterative Individual Verifiability. We now formalize our notion of strong iterative individual verifiability using the following game S-I-IV shown in Fig. 19. This game models a sequential and adaptive attack where the adversary $\mathcal{I}$ must achieve a target, λ_v wins, without a single detection.

To define the outcome of each round in IV, we define another sub-game IV†, which adopts the same modifications as IV* with the one additional modification: IV returns three values: 1 for win, $\perp$ for detection, and 0 for benign loss.

[11] Our threat model assumes voters are honest and will not duplicate envelopes to cause disruption. To mitigate such an attack in practice, physical security measures like unique watermarks on the envelopes could be employed by the envelope printer.

```
Game $\mathsf{S\text{-}I\text{-}IV}^{\lambda_v}_{\Pi,\mathcal{I}}(\mathsf{params})$
1 : $\mathbb{V} \leftarrow \mathcal{I}$ (% Compile electoral roll)
2 : successes $\leftarrow 0$
3 : for $V_{id}$ in $\mathbb{V}$ do
4 :     $r$ = Game $\mathsf{IV}^{\dagger}_{\Pi,\mathcal{I}}(\mathsf{params}, V_{id})$
5 :     if $r = \perp$ then return 0   % Adversary caught
6 :     successes += $r$
7 : endfor
8 : return 1 if (successes $\geq \lambda_v$) else 0
```

Figure 19. Strong Iterative Individual Verifiability (S-I-IV) Definition of strong iterative individual verifiability to account for an adversary's immediate loss if caught.

Concretely, for detection ($\perp$), this means that goal $\neq$ intent, adv_outcome is $\top$ and happy (VoterVerify) is $\perp$.

Theorem 7. *Let Π_{TRIP} be instantiated as described in Appendix L.3.3, and $p_{\max} = Adv^{\mathsf{IV}}_{\Pi_{TRIP},\mathcal{I}}$ be the maximum advantage in the* IV *game. Under the hardness of the Discrete Logarithm problem and the EUF-CMA security of the signature scheme, for any PPT adversary $\mathcal{I}$ playing the* S-I-IV *game with a success threshold of λ_v, its advantage is bounded by:*

$$Adv^{\mathsf{I\text{-}IV}}_{\Pi_{TRIP},\mathcal{I}}(\lambda_v, \lambda) \leq (p_{\max})^{\lambda_v} + \mathsf{negl}(\lambda)$$

Proof. Let Win be the event that Game S-I-IV returns 1. Our goal is to bound $\Pr[\mathsf{Win}]$. Let $R_i \in \{1, 0, \perp\}$ be the random variable for the outcome of the $\mathsf{IV}^{\dagger}$ game against voter V_i. The adversary wins if

$$\left(\bigwedge_{i=1}^{|\mathcal{V}|} R_i \neq \perp\right) \wedge \left(\sum_{i=1}^{|\mathcal{V}|} \mathbf{1}_{R_i=1} \geq \lambda_v\right)$$

The proof proceeds via two lemmas.

Lemma 8. *For any round $i \in \{1, \cdots, |\mathcal{V}|\}$ and for any PPT adversary $\mathcal{I}$, the probability of success in that round is bounded by $p_{\max}$, conditioned on the event that no prior round resulted in a detection, and on the full transcript T_1^{i-1}. Formally,*

$$\Pr[R_i = 1| \wedge_{i=1}^{i-1} (R_i \neq \perp) \wedge \mathsf{T}_1^{i-1} \leq p_{\max} + \mathsf{negl}(\lambda)$$

Lemma 8 Proof We proceed via a standard reduction. Assume for contradiction that there exists a PPT adversary $\mathcal{I}$ for which this inequality does not hold for some round i. That is, for some history $\mathsf{T}_{i-1} = \wedge_{j<i} S_j \neq \perp \wedge \mathsf{T}_{i-1}$, $\mathcal{I}$'s success probability $p_i = \Pr[S_i = 1|\mathsf{T}_{i-1}]$ is greater than $p_{\max} + \mathsf{negl}(\lambda)$. We construct an adversary B that uses $\mathcal{I}$ to win IV game with probability p_i, contradicting the definition of $p_{\max}$.

B receives the parameters of $\mathsf{IV}^{\dagger}$ game targeting an honest voter oracle V. B internally runs $\mathcal{I}$ and perfectly *simulates* the history T_{i-1} by playing the role of the honest voters for rounds $\{1, \ldots, i-1\}$ (i.e., creates a fake history for all previous voters). At round i, B connects $\mathcal{I}$ to the voter oracle V in the $\mathsf{IV}^{\dagger}$ game. From $\mathcal{I}$'s perspective, this interaction is indistinguishable from a real round i of S-I-IV, as the sources of randomness (the voter's choice for n_c from the distribution D_C) are fresh and independent of past transcripts. B then forwards $\mathcal{I}$'s output to the challenger.

B's advantage in its $\mathsf{IV}^{\dagger}$ game is exactly p_i. Our assumption $p_i > p_{\max} + \mathsf{negl}(\lambda)$ thus implies B can break the underlying cryptography underpinning the $\mathsf{IV}^{\dagger}$ game, a contradiction. Therefore, the lemma holds. □.

Lemma 9. *For any PPT adversary $\mathcal{I}$,* $\Pr[\mathsf{Win}]$ *is bounded by* $(p_{\max})^{\lambda_v} + \mathsf{negl}(\lambda)$

Lemma 9 Proof. For the Win event to occur, the adversary must achieve λ_v wins before any $\perp$ outcome occurs, at which point the game terminates. Let's analyze the adversary's probability of success. The adversary's strategy involves choosing which voters to attack. Let the sequence of attacked voters be $V_{i_1}, V_{i_2}, \ldots$. To win, the the adversary must succeed in the first λ_v of these attacks.

Let E be the event that the adversary succeeds in its first λ_v attempted attacks. The Win event is a sub-event of E, so $\Pr[\mathsf{Win}] \leq \Pr[\mathsf{E}]$. We can express $\Pr[\mathsf{E}]$ using the chain rule of probability:

$$\begin{aligned}\Pr[\mathsf{E}] &= \Pr[R_{i_1} = 1] \cdot \Pr[R_{i_2} = 1|R_{i_1} = 1] \cdot \ldots \\ &\quad \cdot \Pr[R_{i_{\lambda_v}} = 1| \wedge_{k=1}^{\lambda_v - 1} R_{i_k}] \\ &= \prod_{k=1}^{\lambda_v} \Pr[R_{i_k} = 1| \wedge_{l=1}^{k-1} R_{i_l} = 1]\end{aligned}$$

For each term in this product, the conditioning event $(\wedge_{l=1}^{k-1} R_{i_l} = 1)$ implies no prior detections occurred. By Lemma 8, each term is bounded by $p_{\max} + \mathsf{negl}$.

$$\begin{aligned}\Pr[\mathsf{Win}] \leq \Pr[\mathsf{E}] &\leq \prod_{k=1}^{\lambda_v} (p_{\max} + \mathsf{negl}) \\ &= (p_{\max} + \mathsf{negl})^{\lambda_v} \\ &\leq (p_{\max})^{\lambda_v} + \mathsf{negl}\end{aligned}$$

□

We now conclude the proof. Lemma 8 establishes a bound on the adversary's per-round success probability, showing that past interactions do not increase its chances beyond $p_{\max}$. Lemma 9 then leverages this per-round bound to analyze the probability of the global Win event. By substituting the final result of Lemma 9, we have:

$$\mathrm{Adv}^{\lambda_v}_{\Pi_{TRIP},\mathcal{I}} = \Pr[\mathsf{Win}] \leq (p_{\max})^{\lambda_v} + \mathsf{negl}(\lambda)$$

This completes the proof. □

This result demonstrates that with TRIP's non-repudiable proofs of malicious behavior, the security against scalable attacks on individual verifiability scales exponentially with the number of votes the adversary must alter, as shown in Figure 17b.

M Extensions to the Base Votegral System

In this section, we present several extensions to Votegral that generally improves usability, verifiability and coercion-resistance.

M.1 Voting History

The use of fake credentials in principle allows voters to see a record of how they voted in recent elections with a particular credential (real or fake). Being able to see later how one voted is an obvious feature often desired in e-voting systems, but is usually not allowed due to receipt-freeness concerns [27]. In the case of TRIP, however, allowing voters to see how they voted in the past does not constitute a receipt or compromise coercion resistance because the record of votes cast *with a particular credential* does not leak the crucial single bit of information of whether this is was a real or fake credential, and hence whether these votes actually counted or not.

From a cryptographic perspective the ability to view past votes does not affect TRIP's security properties such as verifiability: e.g., TRIP remains verifiable even if the VSD does not allow voters to view their past votes. Nevertheless, we feel that allowing voters to view their past votes can improve the *perception* of transparency in the voting process, and in this way can provide a useful psychological benefit towards the acceptance of an e-voting system, and the acceptance of the results of any given election. Such psychological effects of viewing voting history remain to be studied systematically, however.

Furthermore, this approach actually offers a verifiability advantage, as each vote can be accompanied by a receipt proving the ballot contains that vote. Voters could use a second device to verify the accuracy of their vote not only for the current election but also for all previous ones. Additionally, even if a second device is not available immediately, a new device purchased later can verify the integrity of votes cast on the previous device(s).

Voters can also request the election authority to reveal all previous votes cast with a credential stored on their device. To accomplish this, the voter's device proves ownership of the credential to each election authority member and requests verifiable decryption shares of the ballots cast with it. The voter's device then reconstructs the vote using the decryption shares while verifying its integrity. Importantly, this process does not disclose the vote to any election authority members since the reconstruction is performed locally, on the voter's device.

M.2 Reducing Credential Exposure

The voter's real credential is vulnerable in two places: during transport on the printed receipt and at the issuing kiosk To mitigate transport risks, envelopes are designed with a hollow section that conceals the confidential QR codes when the receipt is inserted. A transparent window reveals only the non-confidential second QR code, which contains check-out material, preventing exposure of the entire receipt. At the kiosk, exposure can be addressed using the signing keypair to sign votes when they are posted on the ledger. Voters' devices monitor the ledger to detect if a vote was cast using the signing keypair it holds. Coercion-resistance is preserved because c_{pc} is a non-deterministic encryption of c_{pk} so releasing c_{pk} publicly reveals nothing about whether a particular c_{pc} contains c_{pk}.

However, impersonation creates inconvenience for voters, who must contest fraudulent votes with the election authority. To avoid this, another approach enables voters' devices to sign a newly generate key pair $(\hat{c}_{\mathsf{sk}}, \hat{c}_{\mathsf{pk}})$ with their kiosk-issued key pair. The device then publicly discloses the signature, along with the new public key $\hat{c}_{\mathsf{pk}}$, allowing only votes cast with $\hat{c}_{\mathsf{pk}}$ to be tallied. This process effectively transfers voting rights from the kiosk-generated key pair to the device-generated one. Additionally, it enables voters to port their credentials to new devices, rendering the old device's credential unusable. Both approaches apply to fake credentials since they are also just signing key pairs.

M.3 Resisting Extreme Coercion Scenarios

A significant challenge with remote voting systems is handling voters under constant coercion (e.g., by a spouse). In TRIP, such voters might struggle to hide their real credential from a coercer who could, for example, conduct a search shortly after registration. TRIP can allow these voters to delegate their voting rights to a well-known entity, like a political party, while leaving the booth with only fake credentials. Thus, when their coercer searches them, all credentials found are fake. However, this approach requires voters to trust the kiosk, as voters have no material evidence that the kiosk acted honestly.

We envision the kiosk asking voters if they believe they cannot cast their intended vote outside the booth (e.g., lacking a device outside a coercer's control). If the voter acknowledges, the kiosk prompts them to delegate their vote by selecting a political party they align with. The kiosk then encrypts the political party's preloaded public key (P), which becomes this voter's blinded public credential tag. This process does not require the kiosk to have the private key for P, eliminating any exposure of the party's credential private key to the registrar. Each political party's vote will then be counted for each voter who delegated their vote to that party.

N Side-Channel Attacks and Defenses

While side-channel attacks are beyond this paper's scope and threat model in general, we acknowledge that they represent important and potentially-realistic threats in practice.

The most privacy-sensitive stage of TRIP's registration process is of course the time the voter spends in the privacy booth to print their real and fake credentials. A variety of side-channel attacks could compromise the privacy that the booth is intended to afford. We briefly discuss a few such potential side-channel attack vectors below, together with some potential mitigations. This list is by no means comprehensive, however, and any systematic treatment of side-channel attacks we leave to future work.

In general, a coercer need only learn the precise *number of credentials* that the coercer's victim creates in the privacy booth. Given precise knowledge of this number, the coercer can simply demand that the victim hand over exactly that number of credentials after registration, and will know that one of these credentials must be the victim's real credential, even if it is not obvious which one that is. The coercer can simply vote the same way with all of the victim's credentials, knowing that whichever one is real will cast a vote that counts. Some of the discussion below therefore focuses on side-channel attacks that might give a coercer precise knowledge about the number of credentials created by a voter inside the booth.

For further reference, a broader survey on side channels may be found in [92].

Recording. Perhaps the most-realistic side-channel attack against the privacy of TRIP credential printing is the direct analog of the "ballot selfie" attack against privacy booths used for in-person voting: i.e., voters might be coerced to use recording devices in the privacy booth. We discuss this threat vector and potential mitigations more extensively in Appendix E.

Timing. A coercion adversary C might be able to infer the number of credentials a voter creates by analyzing the duration spent in the booth. Kiosks might introduce a random, artificial delay during credentialing to counteract this threat, making the adversary's estimations more challenging. This delay might appear during screen changes, or a static "Please wait..." message without any underlying activity. Further, kiosks could enforce, for each voter, a minimum random number of fake credentials.

Sound. The noise generated by TRIP's credential printer might expose the of precise number of credentials a voter creates, if the coercer is near enough to hear the printer noise. To mitigate this side-channel threat, we suggest minimizing printer sounds using a low-noise printer. Designing privacy booths as separate rooms with a degree of soundproofing could provide a stronger defense against sound-based attacks, at a significantly higher construction cost of course. Additionally, introducing a noise simulation device might help mask the origin of the sound.

Electromagnetics. A coercer located nearby might use more sophisticated scanning technology to detect electromagnetic leakage from the kiosk the victim is using, or to "see through" the privacy booth's protection in other ways. Privacy booths could in principle be TEMPEST-shielded to protect against such attacks [57], but we doubt the cost of such defenses will be seen as justified by hopefully-rare coercion threats in most cases. As pointed out in Appendix E.2, however, some sites where we envision TRIP might be deployed – such as at embassies or military bases for the use of expatriates and military personnel, for example – might already have suitable areas with stronger side-channel protections, thus reducing the incremental cost of stronger defenses.

Component	**Go**	**Python**	**C**	**PHP**	**Total**
TRIP-Core	1876	0	0	0	**1876**
TRIP-Peripheral	757	0	0	0	**757**
TRIP-UI	0	2901	0	1408	**4309**
Votegral-Tally	1816	0	424	0	**2240**
Total	4449	2901	424	1408	**9182**

Table 4. Lines of code across Votegral sub-systems. C code in Votegral-Tally is to integrate with a Bayer Groth Shuffle C library [47]

O Prototype Implementation Code Size

As a rough proxy metric for implementation complexity, we present in Table 4 a breakdown of the code size of the current prototype implementation. The components are as follows:

- **TRIP-Core**: This component contains the core logic and cryptographic operations as outlined in Appendix J.
- **TRIP-Peripheral**: This component includes the logic required for creating and processing QR codes as well as operating peripheral devices for §7.1 and §7.2.
- **TRIP-UI**: This component includes the user interface logic for the usability study, featuring a web server for managing the user survey and the user interface definitions.
- **Votegral-Tally**: This component comprises the logic and cryptographic operations detailed in Appendix K.

The TRIP-Core and TRIP-Peripheral components are the ones used in this paper for evaluation of user-observable delays in §7.2. The TRIP-Core and Votegral-Tally components are the ones primarily used in our evaluation of the computational efficiency of the "end-to-end" registration, voting, and tallying process described in §7.3. The TRIP-UI component contains our prototype registration user interface; it is currently used only in the companion usability study [97] and is not yet integrated into the rest of the TRIP and Votegral prototype that this paper focuses on.

Groups	(C)ontrol	(F)ake Creds.	(M)alicious Kiosk	(S) & (F)	(S) & (M)
Scales	*System Usability Scale*				
Overall	69.6	70.4	69.9	67.3	62.7
SD	18.6	18.6	17.4	19.8	21.9
N	29	28	29	30	29
Percentile	55.1	57.8	56.1	47.8	35.0
Reporting	*Malicious Kiosk Detection*				
Facilitator	0%	0%	10%	0%	47%
Survey	10%	7%	20%	7%	57%

Table 5. System Usability Scale & Malicious Kiosk Detection The top half displays the System Usability Scale (SUS), including standard deviation (SD) and sample size (N). The Percentile Rank measures the group's usability relative to 446 studies [124] The bottom half presents the proportion of participants who reported the kiosk's misbehavior to the facilitator or on the survey. Security priming significantly increases reporting rate (chi-squared $p = 0.004$) but at the cost of a perceived decrease in system usability.

P Summary of Usability Study of Votegral

This appendix briefly summarizes key lessons from the companion paper focusing on the usability of Votegral [97].

Study Design. The study simulated voter registration according to §3.2, with variations for A/B testing: The 150 participants were randomly divided equally into 1 control and 4 experimental groups: (C) only real credential, (F) real and fake credentials, (M) group F with a misbehaving kiosk, (SF) group F with security education, (SM) group SF with a misbehaving kiosk. The control group only received a real credential known as "voting credential". In group F, participants experienced the anticipated voter registration process: real and fake credentials using a well-intentioned kiosk. In group M, participants encountered an intentionally malicious kiosk, designed to trick them into creating a fake credential under the pretense of creating a real one. Groups SF & SM are identical to groups F & M, respectively, but are shown an instructional video that explicitly states that the kiosk can, in the unlikely event, misbehave. It includes the visual cues of a misbehaving kiosk that intends to violate ZKP soundness. After registration, participants then activated their credential on a mobile device, cast a mock and completed a survey.

Usability. Among the 120 participants capable of creating fake credentials, 92 did so, with 90% of them successfully casting a mock vote using their real credential. After considering those requiring assistance throughout the study, either during credentialing or credential activation, TRIP exhibited an overall success rate of 83%.

System Usability Scale. The companion paper's study [97] used the System Usability Scale (SUS) [35] to measure an individual's subjective assessment of the system's appropriateness for its purpose. SUS consists of ten standardized statements, where participants rate their agreement with them from *Strongly Disagree* to *Strongly Agree* (a 5 point Likert Scale [94]). Table 5 shows each group's SUS score. The average SUS score from 446 studies of diverse systems is 68 with a standard deviation of 12.5 [124]. TRIP (Group F) achieved a SUS score of 70.4, which falls within the 57.8th percentile of these studies and according to Bangor et al. [22], earns an adjective rating of "Good". Furthermore, with the control group's SUS score of 69.6, this defies expectations that participants perceive a usability drop with the addition of fake credentials. We also observe that group SF's score is 67.3, even though the only change between F and SF is the instructional video stating that the kiosk may misbehave. This suggests a potential interesting tradeoff between security vigilance and user experience.

Malicious Kiosk Detection. Given that a reasonable percentage of voters should be able to detect and report deviations from a compromised kiosk, we estimate this reporting rate. We observe that in group M, 10% of participants reported the kiosk's misbehavior to the facilitator, while 20% reported it on the survey. These rates for group SM substantially increase to 47% and 57%, respectively; a statistically significant improvement over group M's detection rate (chi-squared test, $p < 0.01$). These results indicate that voters can, especially when security primed, detect a misbehaving kiosk. Moreover, no participant who interacted with an honest kiosk (C, F, and SF) reported any misbehavior to the facilitator. A few participants reported oddities with the kiosk in the survey, but were related to confusion with the credentialing process.