On the Estonian Internet Voting System, IVXV, SoK and Suggestions

Abstract. The Estonian i-voting experience is probably the richest to analyze; a country that is considered a pioneer in digitizing both the government and private sector since 2001 followed by online internet voting (i-voting) in 2005. However, there are still some complaints submitted, critics and remarks to consider about the IVXV system. In this paper, we introduce a Systemization of Knowledge of the Estonian IVXV i-voting system and propose some added security enhancements. The presented SoK discusses applications implemented by election observers in 2023 & 2024 elections, which, to our knowledge, have never been mentioned and/or analyzed in the academia before. We also point out to unnoticed automated formal verification analysis of IVXV; the researchers discovered a privacy attack that we show extendable to a possible large scale encrypted vote copying. In addition, we identify and analyze recent fixes and improvements in the June 2024 version used in the European Parliament elections connecting them to their academic sources. Finally, we discuss the current system status, propose our own suggestions to some remaining vulnerabilities, then raise the inevitable question of the approaching quantum threat.
Keywords: IVXV, El-Gamal Encryption, Verkle Trees, End-to-End Verifiability (EEV), counted-as-casted, vote buying/coercion.

1 Introduction

Estonia is a small 1.35 million population country located in east Europe who gained independence from the Soviet Union in 1991 and joined the European union in 2004 [1].[1] Most Estonian citizens welcomed the earlier *digital transition*; however, when it came to e-voting in 2005 there were some kind of "*notable divisions within the society between those who fully trust and those who fully distrust internet voting*" as quoted from the ***OSCE-ODIHR*** (*Organization for Security and Cooperation in Europe- Democratic Institutions and Human Rights*) **2023** report [2]. This is reflected clearly in the i-voting statistics; although the Estonian experience could be viewed as the earliest and most advanced [3], the official site [4] shows the ratio of i-votes to reach its maximum of 51% in 2023 (local Parliament) then down to 41.7% in 2024 (European Parliament). One can trace a long history of objection incidences mostly from right wing parties in [1] and [2/(page 8, footnotes 16&17)]; the situation was emphasized in 2023 when internet votes flipped the results for one of those parties

[1] The live number in 30/4/2025 is 1,347,056 (https://www.worldometers.info/world-population/estonia-population/). More detailed statistics, but dated to 8/2024, are in (https://www.stat.ee/en/find-statistics/statistics-theme/population/population-figure#); **1,127,312 "citizens"** :296,268 **~ 26.28% are from Russian ethnics**, (~ 1.35m-1.127m) are ≥ 1 yr residents.

(***EKRE***). Analysts view it as a natural echo of the society division mentioned above; i.e., it is expected for the curves, [5], showing the distribution of internet votes to be completely different than that for paper votes according to each party's preference. Still, there were some complaining activities that continued persisting to the 2024 European elections [6].

To complete the picture, it is appropriate to note that Estonia (with ~26% citizens from Russian ethnics[1]) has Ukraine as its closest neighbor, and according to [7] its i-voting system has faced Russian attacks that authorities say were properly defended. Since most security metrics are probabilistic [8/sec. 4], the power and incentives of possible attackers has an impact on the probability of an attack to succeed; whether it is just a party competing for a win, or a powerful country that has farsighted interests. However, we here just notify of a possible impact and leave the analysis of measuring/estimating it to political science researchers.

After this brief preface on the Estonian election environment and the involved players, we proceed into the technical and cryptographic details; hence, the rest of the paper is organized as follows. Section 2 reports some recent important activities by the i-voting opposing community that have technical merit, while section 3 marks briefly the milestone steps through the evolution of the Estonian i-voting system[2]. Then section 4 explains in detail the current version of IVXV ending with an important attack that was fixed before the 2023 elections, and section 5 goes through recent improvements that were made in IVXV before the European elections. Section 6 discusses the remaining vulnerabilities of the system along with suggested solutions that addresses the concerns of EKRE as stated in the OSCE 2025 report [9/sec.3.3/28] and ends by discussing the quantum computing threat and efforts toward it. Finally, section 7 introduces suggestions for further future research and section 8 concludes the paper.

2 Recent Opposition Activities with Technical Merit

A technical incident that gained some publicity in 2023 elections [10] was done by the same computer scientist observer[3] in [4]; he ***voted using his own Python code*** [11,12]. This gave an alarm that the voting application, which voters should download to deliver their vote, is not authenticated by the system; the *OSCE* report [2/page 8] believes the incident "*could present a cyber security risk*". The report also mentions *some wrong district votes*; the incident was magnified by the opposition [13,14], described by IVXV representatives [8/ sec. 5] as a population registry problem that was quickly detected and resolved due to the i-voting.

[2] Since there were many previous studies discussing earlier years of the Estonian i-voting, like [1] till 2019, this paper is more concerned about the vulnerabilities and improvements of the recent years. Our title says IVXV; a design that took place in 2017.

[3] The word "***observer***" is used by IVXV to acquire certain access rights during election (as opposed to "auditors"). The term "***Computer Scientist***" (taken from [9]) is rejected by IVXV representatives who describe Märt Põder as "***A hobby hacker activist***", while the OSCE report referred to him as "***someone with sufficient programming skills***"

Even if some exploits and/or problems were magnified, it is the authors' impression that most complaints get rejected without objective investigation based on a submission deadline (*3-days from election*); [15] is a very recent (20/6/2025) example. The OSCE 2025 report comments on this issue by saying [9/sec. 3.3/68 & Recomm. J] "*the legislator should also address the specific complexities of internet voting when determining appropriate deadlines"*. To be fair we noticed that in many cases, as will be illustrated throughout the paper, the vulnerability gets handled and fixed silently before the following election.

We also report another rejected complaint about ***the decryption of invalid votes*** (after the 2024 European Parliament elections), but the rejection was done objectively this time [16]. Among the three listed reasons, being an *observer* not an *auditor* seems to be the dominant one, where auditing is organized by the State Electoral Office in all elections. According to [17/Conclusion], generating proofs of correct decryption of invalid votes was remediated in code before June 2024, but *the file containing the decryption of invalid votes is only accessible to auditors* [17/page 22]; this will be further discussed in section 5.1.

The recent European parliament elections were accompanied by some newer activities from the i-voting opposing community [5,15,16,18,19[4]]. The same observer mentioned above has developed some kind of ***shadow e-voting*** site called *virtual threshold survey* [18] encouraging citizens to vote again on it as a check (although no evidence of considerable participation ratio).

Also worth mentioning is a complaint about ***the desktop used for keys creation at the setup phase*** not being completely isolated and has some extra software downloaded onto it. A first earlier complaint granted an observer (23/2/2023) permission to see the content of the backup copy of the boot hard disk used in key creation to have full confidence there is no malware in the computer memory during key creation [20]. The observer took the photo shown in Fig.1 when the disk inspection took place (28/11/2023), then pursued the matter further to the supreme court. The supreme court responded that this could not have affected the voting results; quoting their exact words, [21], "*voting results cannot be compromised with malware, because with the help of the reading certificate issued when determining the voting results, the compromise would be revealed immediately*".[5] On Aug 2024, a commentator from IVXV team stated (in an earlier reviewers' report of an earlier version of this paper) that they admit the risk and "*it has been taken care of*".

[4] Prof. AGO Samoson is an NMR & materials science researcher in Tallin University of Technology (TUT) (https://www.researchgate.net/profile/Ago-Samoson); as IVXV representative clarified, "*Cybernetics* is NOT the same institution as *Cybernetica, it shares some common history, but this ended more than 25 years ago and the researcher Ago Samoson is in no way affiliated with Cybernetica, nor IVXV development*", where *Cybernetica* is the company behind Estonian i-voting system partnering with *Smartmatic*.

[5] The official progress of events is in (https://www.riigikohus.ee/et/lahendid/?asjaNr=5-23-40/2)

Fig.1. An image taken from [20]; according to the observer, this is the computer used in key creation which was supposed to have only an authentic Windows10 operating system, but he detected other software installed on it; DigiDoc4, Notepad++ and RamDisk.

Finally, another scientific report from ***the Cyber Security Committee of the Academy of Sciences*** has been handed to the election organizers [22]. The complete report is not published, but the minutes of the committee meeting [23] on 3rd of June 2024 clarifies they *have identified* ***6 threats*** *whose risk class is higher than small*[6]. We chose to mention the report here, although it is not an opposition activity, to illustrate that there are some problems; i.e., and not all opposition complaints are exaggeration.

3 Earlier System Evolution

As mentioned earlier, digitization has been in Estonia for more than 20 years, even before 2001, and was extended to include the private sector hand in hand with the e-government; e-ID cards existed since 2002, and electronic transactions is the casual behavior of the Estonian citizen. Details on digital system architecture and components like *Xroad*, *KSI* private blockchain, [24], seem irrelevant here. However, we find the e-ID key generation relevant since it is used in internet voting from its beginning in 2005. Hence, we will dedicate section 3.1 to one major event that changed a core cryptographic component of the e-ID system, **RSA;** then we will follow with a brief overview of i-voting earlier evolution till it reached its main design as IVXV in 2017.

3.1 Electronic Identity Card 2018 problem

In May 2018, Estonian authorities officially declared a persisting problem that started to appear in some rare incidences of duplicate RSA keys since 2011/2012. Citizens with problematic keys were asked to re-install the Java Applet on their cards at PPA (Police and Border Guard Board = *Politsei-ja Piirivalveamet* in Estonian language) stations. When reinstallation became more frequent, researchers started to analyze the accumulating data for the root problem. Then, it was proven that the ID card

[6] A system representative commented (14/8/2024) that all the 6 threats are of risk class medium (11-13); we have no evidence on that except a previous review report of this paper.

manufacturing company, ***Gemalto***, generated the RSA keys outside the chip, which violates the agreement rules and gives a chance to copy and/or repeat key pairs. A lot of interesting details on how the analysis was done can be found in [25]; more faulty keys issues[7] can be found in the same researcher's, Arnis Parsovs, PhD [26], and independently in [27]. Also, other RSA vulnerabilities were discovered in [28] which justified aborting RSA as an algorithm, not just the company.

According to [1], this was a global crisis for the company which was sued in many other countries. Spain and Slovakia [29] replaced all the physical cards, while Estonia fixed them remotely; then changed the company to ***IDEMIA*** [30] and, as recommended in [25], moved to threshold cryptography and homomorphic encryption. Currently, the Estonian i-voting system IVXV, [31], also uses ***384-bit Elliptic Curve*** Cryptography ECC with El-Gamal Encryption (section 4); yet, the list of authorized votes is still signed using 2048-bit RSA key, maybe that's why [32] nullifies the effect of eID problems on IVXV. Note that both IVXV and all digital IDs will have to migrate into post-quantum cryptographic algorithms on the near future.

3.2 Estonian i-voting before IVXV

As a preface, this section gives a condensed brief on how the Estonian i-voting system has evolved from 2005 to its final form as IVXV.
The main design theme since 2005 is a *double envelope protocol* sending voter signed ballot after first encrypted by the election public key[8] to the vote collector. Then, we mark 2 milestone step transitions [sec.1 of 33,34,35]:

In 2011, a student named *Paavo Pihelgas*[9] demonstrated a proof-of concept ballot-manipulating software that relied on the absence of an acknowledgement to the voter that his/her vote was received. Hence, [36], *the ability for voters to verify their votes* was first introduced in 2013. However, several flaws were discovered in 2014-2016, [37], that could maliciously alter the vote or the QR code.

Then, in 2017, Cybernetica partnered with *Smartmatic* to produce a new design, IVXV [38], with the QR verification code in its current form. A lot of improvements on IVXV since 2017 include a *vote-registration* service to guarantee no vote dropping, a *shuffling re-encryption mix-net* that cryptographically shuffle anonymized vote ballots to guarantee vote privacy, and a *Schnor based non-interactive zero-knowledge proofs (NIZKPs)* to prove the shuffled mix-net output decrypts to the same value as its original input; all of this will be detailed in the next section.

---

[7] Including codes printed too dark that they were readable using torch, without opening envelope (happened in 2002 with the old company then again in 2018: https://news.err.ee/886313/new-id-card-issue-codes-can-be-read-using-torch-without-opening-envelope), duplicate email addresses in certificates, issuing certificates with incorrectly encoded public keys, failing to revoke certificates of deceased persons.

[8] That's why it is called "*double envelope*"; the vote is embraced with 2 cryptographic keys: the voter key and the election key.

[9] The student filed a complaint, [34], to the Estonian Supreme court requesting to nullify internet votes in 2011 elections. The complaint was dismissed for passing the 3 days limit (https://www.riigikohus.ee/en/constitutional-judgment-3-4-1-4-11).

4 IVXV

In this section, we explain the design and structure of the Estonian internet voting system, IVXV, as described in the official documents [39]. Then we detail an important cryptographic attack along with its fix before the 2023 elections.

4.1 Brief Factsheet

The developing companies are ***Cybernetica-Smartmatic*** [3,38]; the voting device must be a desktop PC; voting can be done using *mobile-ID*, *Smart-ID*, or any digital identity integrated in the *web-eID*[10]; *multiple voting* is allowed to avoid coercion or vote buying (only last vote is counted and a poll station vote overrides all i-votes); ***El-Gamal Homomorphic*** Encryption scheme is used to encrypt votes then the encrypted vote is digitally signed by the voter (double envelope); optional vote verification can be done by voters (through *QR codes* using a second mobile device) within 30 mins of voting with a max of 3 times; ***Mixnets*** are used to scramble votes before decryption to preserve ballot secrecy. The setup phase, before the election starts, constructs the *election secret key* from 9 parts issued by the members of *the Election Commission of the Republic*, such that decryption requires *5 out of 9 parts*[11]. Finally, after the election, there is ***an auditor application*** (could be run by anyone) that verifies the cryptographic NIZKP proofs provided by IVXV on the election published output data.

4.2 System Architecture & Voting Steps

The system architecture and voting steps are depicted in Fig.2, which could be summarized as follows

1. The voter installs the voting application, sometimes abbreviated as **VA**, on his/her PC.
2. After submitting the digital identity ID, the voting application sends to *the vote collector*, **VC**, which in turn sends to *the registration service*, **RS**, to check the eligibility of the voter to vote; if eligible replies with the candidate choices for that voter (according to district) to be displayed to the voter.

[10] The newer IVXV version for EP-2024 included extra ***web-eID assistance service***, ***Smart-ID assistance service***,…to scale horizontally enabling the usage of different digital identities (section 2 of the architecture, sections 8.5-8.6 of the protocols in [41]). The web-eID solution (https://www.id.ee/en/article/web-eid/, https://github.com/web-eid/web-eid-system-architecture-doc) is part of the European Union web-eID project for all public key cryptography digital identities across Europe.

[11] Hence comes the term "***threshold cryptography***" in the end of section 3.1, because a threshold is agreed upon from the beginning (here 5 out of 9); i.e., if ≤ 4 committee members were malicious, they cannot collude to leak the key. Note however, that this does not prevent the key from being leaked through a compromised storage media like the desktop of Fig.1 for example.

3. The voting application encrypts the voter choice using the election public key (El-Gamal encryption), adds the user signature on the encrypted vote (with the voting application running on the voter's PC and after the voter's approval, *the voting application has the right to sign a message with the voter signature*), adds also the signed *timestamp certificate*[12] received from the registration application through the vote collector *after verifying the signatures of both*, and then sends the double envelope ballot to the vote collector.
4. The vote collector application validates the voter's signature; after validation, the signature is removed, and the encrypted vote is added to the list of votes stored in the *Ballot Processor*. After voting is closed, and before sending ballots to the mix-nets, the ballot processor performs some integrity checks, removes multiple votes and votes overridden by poll station voting. Then the remaining list of "to be counted votes" goes through mix-nets to hide their original order and verifiably decrypted[13] at the counting phase.
5. The vote collector sends a verifying ***QR code***[14] to the voter for optional vote verifying (through verification application) using a second smart device.

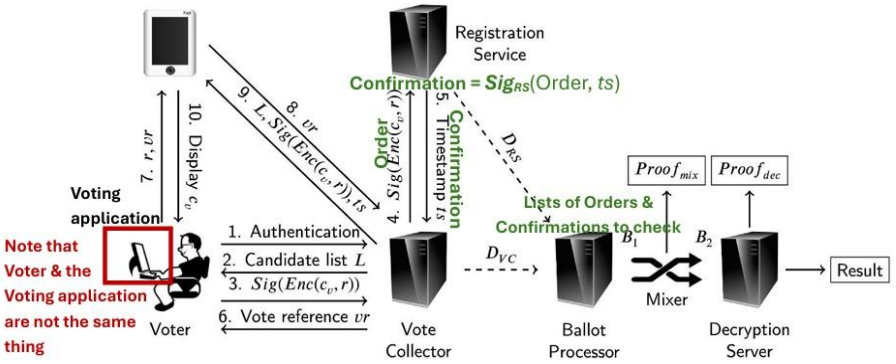

Fig.2. A diagram describing the architecture & the steps of the Estonian voting system; adopted from [1] with adding some remarks in red, and new updates in IVXV 2024 in green

[12] When [1] was written, the timestamp certificate was used only to distinguish the last vote and to check the 30 mins verify duration. In 2024 version, we will see why in section 5.3, *the certificate* sent by RS to VC *is* a ***signed CONFIRMATION*** that contains the original request (***ORDER***) sent and signed by the VC, along with the *timestamp*.

[13] IVXV uses *Douglas Wikström's Verificatum* (https://www.verificatum.org/); the package provides a verification application, and IVXV too (and several other projects [33])

[14] According to [11], there was a revealing incident of *the president vote* through his QR code: he showed it in front of cameras to encourage citizens and someone took a snapshot. As mentioned in [8], this is not considered a privacy exploit since the President voluntarily revealed his QR-code.

4.3 Cryptographic Details of Last Fixed Attack

The exploit introduced in 2022 by [34], and fixed in 2023 by IVXV, give us a closer look into the cryptographic details of El-Gamal encryption algorithm; especially since we will mostly treat it as a black box in the rest of the paper.

-Let the election public key be "**y**" with corresponding secret key "$\mathbf{S_k}$", and "**g**" be the generator for El-Gamal encryption; hence, the equation « $\mathbf{y= g^{Sk}}$ » holds.
-To encrypt a vote "**v**" the voting application generates a random number "**r**", so that the encrypted vote is $\mathbf{(C_1, C_2) = (g^r, y^r v)}$
-The verification application, working instantly within 30 mins, receives "**r**" from the voting application (hidden in the QR code) and calculates $\mathbf{v = C_2 / y^r}$ where the voter is assured when the displayed "**v**" is the same "**v**" he/she voted for.
-When counting votes, the election authority uses the election secret key ($\mathbf{S_k}$) and El-Gamal encryption known equation « $\mathbf{y= g^{Sk}}$ » to calculate $\mathbf{v= C_2/((C_1)^{Sk})}$
-In the older design, the verification application only received $\mathbf{C_2}$ from the vote collector. This gives a malicious voting application the chance to manipulate the encrypted cipher text by sending different values of $\mathbf{C_1}$ for the same $\mathbf{C_2}$. Without checking $\mathbf{C_1}$ value, the verification application will not detect a fraud if the voting application sent a wrong "***r***" value to the vote collector, ***r'*** such that $y^r v = y^{r'} v'$ to deceive the vote collector into recording ***v'*** as the voter's intended vote.

Long story short, the authors found *three possible manipulations* all with *a simple fix:* making the vote collector send the whole encrypted pair (C_1,C_2) to the verification application which should also *verify that* $C_1 = g^r$ as was finally done [40/lines 77-83 & 141-146 & the exception is thrown at line 60] on 23rd Feb 2023 just before March 2023 elections. The authors alarmed that it is concerning [34/sec. 3.6] that such a straightforward vulnerability wasn't noticed earlier and then criticized the quality of IVXV in general [34/sec. 4]. Hence, we add another alarm that the risk remains for nonverifying voters with either a malicious voting application or a malicious communication network.

5 Enhancements Against Vulnerabilities

In addition to extending IVXV to support voting with more kinds of digital identities like web-id and mobile-id, [41], the IVXV version 1.9.10-EP2024 included many other improvements. This section discusses three important fixes that were committed to the official code site, [42], on 30th May 2024, just before the European Parliamentary elections on 3rd of June and research efforts that led to them; Table 1.

Table.1: Fixes/improvements done in IVXV 1.9.10 EP2024

Risk	Deployed Solution	Remaining Issues	GiHub File	Corresponding Academic Research
Invalid votes	Decrypted in a separate file with ZKPs of correct decryption	-Files are viewable by auditors only (complaints) -Better deploy ***Range Proofs*** to prevent invalid votes from entering list.	Embedded in [42] ***DecryptTool.java***	Tallinn Univ. Ms. thesis [17] (Jun 2024)
Ballot Processor (BP) manipulation	Consistency checks on SHA256 hashes of totals and subtotals.	***Offline*** checks; i.e., count based validation depends on trusting the Vote Collector (VC) and Registration Service (RS) to not collude before the list enters the BP	A new file [45] ***IntegrityTool.java***	Tallinn Univ. researchers [46] (Dec 2024)
Timing attacks	Checking *Session ID* and *Timestamps* difference, which are generated by *PKIX* protocol	Cannot detect fast attacks that can manage to work in the duration of one session (like ***Pereira attack*** [35])	A new file [50] ***client.go***	An extension, [48], to a Luxemburg Univ. PhD on formal verification of i-voting systems, applied to IVXV (Jun 2024)

5.1 Decryption of Invalid Votes

Invalid votes are now thrown in a separate file and ***ZKPs (Zero Knowledge proofs) are generated for correct decryption of invalid votes as well***. However, election observers are not allowed to access this file or verify those proofs and there were a lot of debating and complaints about that (section 2); the issue of observers' rights persists in OSCE 2025 report [9/sec. 3.3/65].

Why not reveal invalid votes?

The reason is better explained scientifically as T. Kraavi did in his master thesis, [17], than by the state election service as recorded in the supreme court decision [14].

-Reasons 1&2 in [14] talk about the technical infeasibility of decrypting invalid votes after the election and how this needs parts of the secret election key. Well, this was already done (feasible); [17] details how, and it can be traced in ***DecryptTool.java*** file [42/lines146,164,182,247,277,298].

-While the 3rd reason in [14] of "*not knowing in advance what the invalid ballot contains and it may be an attack*" seems vague and not convincing, the argument is rationalized in [17]; it is the possible reveal of some information about the voter of the invalid vote, or more severe the threat of ***encoding attacks*** described in [43/ sec. 3.3 & 4.1] where an adversary can know the votes of several voters if able to submit a carefully crafted invalid vote and also view its decryption. Hence, the rational is to shrink the circle of trust into auditors only, which is not even needed if invalid votes were rejected earlier by the vote collector as [17] suggests. In fact, tracing the ***number of invalid votes*** in the official statistical site [4] to be ***exactly 1***[15] in the last three local

[15] except 2 in the European elections.

elections since 2021 makes it look quite suspicious; the doubt includes anyone who can see the votes and could be eliminated completely if range proofs were deployed.

5.2 Integrity Checks

Another problem that was mentioned in *OSCE* 2023 report [2] is that the authorities are assumed trusted regarding *the deletion of multiple votes or ill-formed vote ballots*. Quoting their own words "*The critical step of removing the votes overwritten by another vote cast on the internet or in a polling station was not audited*", "*An insider with sufficient resources to alter the system, if able to do so undetected, could manage to control which votes are removed and therefore partially impact the results*". This was viewed by [44] as trusting the vote collector (VC) and registration applications (RS) to not collude, otherwise it would be possible to drop ballots; the *Ballot Processor* in Fig.2 could also manipulate the ballots (assumed trusted by the system). To elaborate more, yes there are decryption proofs that what goes into the *mix-nets* is exactly what gets out of it to be finally decrypted, and yes there is the possibility to design a public independent decryption proof verifier [33], but there was no cryptographic proof for the transition from the total list of votes to the "to be counted" list of votes; what is called the *processing stage* and we believe is part of *universal verifiability*.

IVXV Adopted Solution

What we believe is a partial protection from this vulnerability (see section 6.4), was integrated into the audit application of IVXV through the file, [45], ***Integritytool.java***. However, the details of the solution were only published academically in [46] as of 23/12/2024. The authors state that they have contacted IVXV team with their proposed checks to detect insider risks at the processing stage, and that it was successfully deployed.

The proposed checks are applied to the Ballot Processor data which is an offline computer [46/sec. 3.A] that performs some processing on this data (step 4 in section 4.2) to then input the list of anonymized votes to the mix-nets as shown in Fig.3.

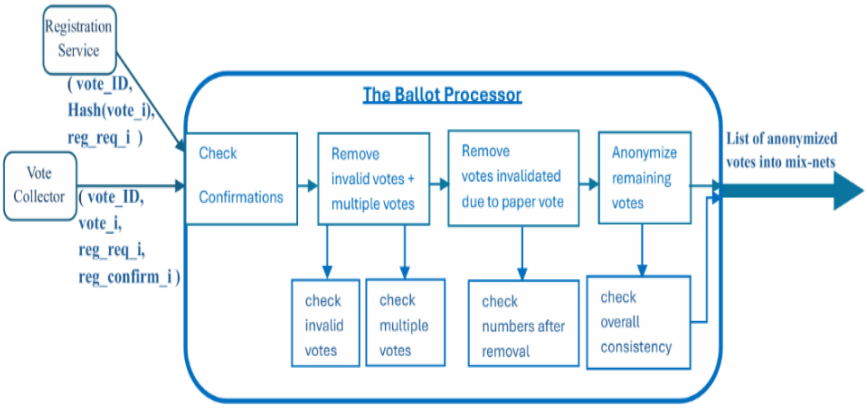

Fig.3. The (modified) processing stage after adding the *integritytool.java* file

Since the votes file (along with the necessary checksums) is cryptographically signed after each step, most examinations depend on comparing SHA256 hashes of subtotals, totals, and individual votes before and after each step. Also, *count-based validation* was needed to detect certain types of manipulations like adding the removed older multiple votes to the list of anonymized votes; i.e.,

Count (original votes file) - Count (anonymous valid votes) =
Count (multiple votes) + Count (replaced by paper) + Count (invalid votes) …Eq(1)

Note that the Ballot Processor receives the list of all votes along with any necessary confirmation checks [41/protocols.pdf/Fig.6.3]; i.e., the integrity of those cryptographic checks (like the *Count (original votes file)*) depends on trusting the vote collector and the registration application to not collude [44]. Hence, what was ***mitigated*** by [45,46] is ***the risk of a colluding Ballot Processor***.

5.3 Session ID & Time Checks

A possible attack by a malicious voting application that could deceive even verifying voters was discovered by Olivier Pereira in [35]; a malicious application could fake a system crash to deceive the voter to vote again. By doing so, the application can take the voter signature twice (generate another "*r*" value to construct and sign a new encrypted vote in the background); hence while showing the voter the QR code of his/her choice, the system will consider it an old vote and will use the new vote.

Previous Suggestions

-The author suggested a few mitigations, K. Krist's PhD [32/sec. 5.3] suggested others; none was actually adopted, and we will mention them as we go.

-One may also recommend advising voters to double check the number of voting transactions with other available e-government services available in Estonia like *myID* service [47], especially if their device has suffered a system crash while voting.

-Both Pereira [35] and [48] suggested verifying the last vote only, while we thought of adding a flag to the QR code on whether this is the last vote or not.

-While adding a timestamp to the QR code won't help[16], a voter attempt tracking number as suggested by [35,48] will change in the revoting screen after the faked crash; however, some may view asking the voter to trace numbers as decreased usability. Also, [8/sec. 4] argues that all such solutions could be used by a coercer/vote buyer to find out if the voter voted again; yet a coerced voter is only at this risk for 30 mins (the verify duration).

-Another simple safeguard from this specific problem, [49], would be *to force a time interval between votes;* the verification interval*, 30 mins,* seems a suitable choice. This must be accompanied by heavily warning voters to close the application then reopen again; if the voter eID remained available on the voter's PC more than 30

[16] Even if the voter noticed the small-time difference, could think "maybe the system records the time when I started, not when I finished"; a vote counter can't reveal following votes either.

mins, a *Ghost Click attack* becomes possible [35] and a malicious voting application would have enough time to submit without voter's knowledge.

Formal Verification & Extended Pereira Attack

Published on June 2024, [48] introduced for the first time an automated formal verification of IVXV security (treating cryptographic functions as black boxes). Their tool rediscovered the Pereira attack and another timing attack that we believe is a generalization of *Ghost click* attack mentioned in [32/ref.166, 35]; the adversary stores the fake vote to submit it as late as possible to guarantee its count. We believe IVXV deployed the time difference & session ID checks suggested in [48].

IVXV Adopted Solution

A solution to the generalized attack in [48] was added on 30th May 2024. The code was updated to check the session-ID is still the same before generating the verifying QR; the code documentation [50/lines 22&103] highlights that this "***prevents reusing session ID until it is deleted from a database or expired***".

Tracing a little deeper, [41/protocols PDF/sec. 8], the ***PKIX*** *(Public Key Infrastructure X.509)*[17] timestamp protocol is used by the registration service to record the time of casting the vote, while the ***rsyslog*** service records the logging time in milliseconds which make it possible to use the *Guardtime* module to ensure the integrity of the logs.

Pereira Attack is Still Persisting!

However, as [48] warns, this solution cannot handle the original attack in [35], since it depends on the fast sequencing of the fake crash-restart scenario; this allows a malicious voting application to use the same session ID twice and have a reasonable time difference. Hence, they suggested the voting application should generate a session tracking number (roughly reflects the number of votes) to be added to the ballot, checked to be new each time and then embedded in the QR-code. The voter then should check that the tracking number on the voting screen equals that on the verification screen; we do not think IVXV deployed this solution.

Before getting access to [48], we asked *grok* (Twitter, X, AI companion) to trace the whole IVXV code for checks of session ID & timing; its reply also confirmed that the threat remains to exist. The two methods *SessionStatus.Read* and *verifyStatusReadResp* can detect a fake restart if the server expires and revokes the same Session-ID afterwards; the current code ***relies on timestamps expiration***, so a malicious application reusing an unexpired Session-ID could slip through unless tighter server-side tracking is in place.

---

[17] ***PKIX*** is a timestamping protocol that enables a trusted third party (called *TMS* in [48]) to confirm the existence of a specific data at a specific point in time with its signature; can check 4 times *thisUpdate*, *nextUpdate*, *producedAt*, *revocationTime* (https://datatracker.ietf.org/doc/html/rfc6960, https//link.springer.com/referenceworkentry/10.1007/0-387-23483-7_302)

6 Remaining Vulnerabilities/Issues

This section starts with a preface that summarizes the status quo of the Estonian electronic voting systems, then follows with some remaining vulnerabilities and proposed mitigations (whether by the authors or in previous literature). As we expect IVXV is approaching the end of an era and will have to face the quantum computing threat very soon; we will discuss the post-quantum applicability of our suggestions as we go, then seal with possible post-quantum system upgrades; Table 2.

Table.2: Remaining vulnerabilities/risks in IVXV 1.9.10 EP2024 and suggested solutions

Vulnerability	Risks/threats	Concerns/Complaints about the issue	Suggested Solutions	Proposed by
Invalid votes	Privacy attacks [43]	Many persisting complaints for viewing their decryption files [14], concerned OSCE/ODHIR too [9]	Deploy ***Range Proofs*** to prevent invalid votes from entering the ballot list at all	Tallinn Univ. Ms. thesis [17] (Jun 2024)
Authenticating the Voting Application (VA)	-Pereira attack [35] -Copy attack on Privacy [48] -Large-scale vote buying/coercion through *encrypted copy attack* + PC execution attests + online coding to automate execution [87,88] -Variety of malicious VA risks	-Cybernetica supervised PhD [32/sec.5-6] -Olivier Pereira [35] -OSCE/ODHIR 2023 report [2] -Many other researchers including the authors of this paper.	Using a microcontroller voting device	Tallinn Univ. PhD [32] (2022)
			-Optional checking of file hash in an Electrum Bitcoin wallet style [55], but batched into 1 click [56] -Assigning a signature key for VA, and allowing optional registering of other VAs but after scanning the code for malicious activities (more robust, but require flexibility and cooperation from authorities to not reject unobjectively)	This paper
Insiders' Trust	VC and RS are trusted to not collude; their collusion may result in: -privacy attacks [43] - different possible manipulations of the ballots list before entering the ballot processor	-Estonian parties and i-voting opposing communities in general [1,2,13,19] -Detected by automated formal verification tools in [48]	-Adding a ZKP to each vote.	[43] (2022)
			-Performing different consistency check queries, and RLAs, between ballots list and other services recording digital transactions in Estonia, like myID [47]. -Using Verkle Trees [64] to cryptographically prove count values.	This paper

6.1 Related Work & the Status Quo

Many academic research papers, in addition to pointing out some attacks, [32,34,43,44,48] have introduced a holistic criticism to IVXV; [32] was concerned with the tradeoff between integrity vs coercion resistance, and handling a corrupted voting device; [32/sec.5.1.2, 44] discussed IVXV trust assumptions including software components and key holders, [44] also analyzed its *public information* as *satisfying* only *1* (minimal restriction on disclosure of vulnerabilities) *out of 9 quality*

metrics[18] and warned from the possible existence of hidden vulnerabilities; [43] demonstrated (through introducing possible privacy attacks) that *IVXV is vulnerable to attacks against vote privacy in those threat scenarios that were considered for it originally*.

The most recent is [48] used automated formal verification methods [51] to discover new attacks against verifiability (see section 5.3) and against privacy using some form of replay attacks [48/Fig.1.D][19], then analyzed the system End-to-End-Verifiability (EEV) and privacy under 9 different combinations of trust assumptions [48/Table1]. Their results show failure if the adversary controls more than some voters' credentials; however, like [34,43,44], they were studying earlier versions of IVXV. We could roughly map the current IVXV version to their EEV^{+} which fails for nonverifying voters and/or if 2 of (VC, RS, timestamping service TMS) colluded; this resembles what we argue here, and try to improve on, through analysis.

We traced many improvements in IVXV 1.9.10-EP2024[20], meaning we could have missed some. However, we know the newest version is the one analyzed in [21,22] which identified *6 threats with risk level higher than small*; the report was done in collaboration with the election authorities (i.e. not biased against or missing updates).

Although the IVXV team sticks to the fact that there was no proved error in the election results, we point out to the low *QR verification ratio of 5.5-9.9%* [4]; this does not really prove beyond reasonable doubt that no pairs were manipulated. Besides, the tone of their statements is more offensive than scientific; examples include [8] to OSCE, and [14] discussed in section 5.1.

Another general problem is that IVXV neither welcomes communication nor advertises its progress to researchers; the authors of [34] were not informed of the fix in section 4.3 till their paper was published; the authors of [48] seem to lack any contact with IVXV and had to work on earlier versions. The improvements in section 5 were discovered through repeatedly scanning the academic literature and tracing the code; the official English site, [52], still have the June 2024 version files only in the Estonian language and we had to dig our way through browsers translators.

6.2 The Voting Application

The IVXV solution discussed in section 5.3 does not protect from the Pereira attack; in addition, it is not a general solution to the voting application vulnerability. The voting application is the only unrevealed part of IVXV code, [32,34], as an open source; [44] identify this fact as a trust assumption. What is more of a threat is the

[18] Used quality metrics were defined in an earlier paper by the same authors (FC'21, *New standards for e-voting systems: Reflections on source code examinations*)

[19] The attack copies b=(C,s) of an honest voter into b'=(C,s') where s,s' are the signatures of "C" the ballot as a whole; they say this strategy of minor biases in the outcome could be generalized leading to quantifiable privacy violation [48/ref.44].

[20] GitHub shows 897 changed files with 34,059 additions & 10,830 deletions. Translating [41], a lot of work was done in integrating different kinds of IDs and in coordinating with XRoad service (X-tree). A whole section is dedicated to Registration Service, [41/protocols/sec.6]; the interaction between online (RIA), offline (RVT) and other IVXV services.

incident of overriding it in 2023 election, [10,11]; accordingly, the *OSCE* 2023 report [2] notified about the risk of not authenticating the voting application. A clear obvious vulnerability comes from *the possibility of downloading a* ***malicious voting application***; this leaves it as *an open challenge for adversaries* to design the most possible malicious code they can come up with. Having a ~ 90-95% probability that the voter will not verify the vote, an adversary could use social engineering to target those who are not likely to verify which increases the risk level.

Another possible risk is *for* ***vote buyers/coercers*** *to do what the authorities haven't done*; i.e., develop a fixed candidate voting application and authenticate its usage through *execution attestation* on the voter PC[21] before transferring the money. *DarkDAO*s risk was discovered in 2018, [53], implemented in 2023, [54][22], as a possible threat to DAOs (Decentralized Autonomous Organizations) where voting on important decisions is automated through an online code; so, it could happen here too.

Combining this risk with the privacy attack[19] in [48] could lead to a kind of massive[23] vote buying/coercion that is not applicable to paper voting or poll station e-voting; ***voters could be forced to copy a vote they do not even know***. Imagine a scenario where an adversary controls "*n"* voters with a malicious VA' as in [53,54]; the adversary either perform an auction between corrupted candidates, or even worse works for another country (recall [7]), then sends an encrypted choice $\mathbf{(g^r, y^r.v)}$ to VA' to copy. If you think you can search the anonymous ballots for suspected repetitions, VA' could add a fresh randomness to each vote $\mathbf{(g^{r+rnew}, y^{r+rnew}.v)}$.

A general solution to all the above is ***to authenticate the official voting application***

-A simple moderate safeguard is to publish its *file digest* (hash SHA256 for its code for example) and encourage voters to run a check before using it. The *Electrum* Bitcoin wallet already gives this option when downloading [55] and the steps could be easily batched into 1 button click as *grok* showed us [56]; this is more robust and usable than just depending on the OS verifying a developer key signature [8/sec. 4]. Although this solution is still voter dependent, we believe one can count on users pressing a recommended button with a higher ratio than the ≤10% QR verifying.

-As a second safeguard for QR-code checkers, the verification application could also display an extra message with the vote saying that "*You voted using (the official/a*

[21] Remote execution attests were originally discussed on Intel SGX, and are available on many PCs(https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions-processors.html), also supported by other vendors.

[22] The same authors next participated in (https://medium.com/initc3org/complete-knowledge-eecdda172a81, https://eprint.iacr.org/2023/044,) which suggests that a TEE only provides execution attests after submitting ***proof of Complete Knowledge (CK***) of the user device

[23] The observable anomalies discussed in [32] and [32/ref.166] to exclude the possibility of large-scale application of compromised voter device attacks in not applicable here; besides, this was analyzed in 2014 before smart contracts [88/sec.3] appeared providing online code execution.

different) voting application". This fulfils the system philosophy[24] of giving suspicious voters the freedom to code and use their own voting application (or maybe one written by a political party they trust).
-Another solution we believe is more intact (since it is not optional) is to assign *a signature & authentication key pair* to the voting application like the rest of applications in the system. Then the election authority can allow only usage of *a pre-registered private voting applications* with a *stored public key* at the voting server; this way, the election authority can also *scan* the private voting applications for any malicious or vote buying code before granting usage rights. However, another issue remains in this solution in ***how to inform a non-verifying voter that the vote was rejected because he/she has installed a malicious voting application***.

We think *the vote collector application could deduce the IP address* of the voter machine from the first contact with the voting application, then it can *send a* direct warning message; something that pops up on the voter screen "Be careful, this is not the official voting application". The use of *a feedback channel* was also suggested in [32/sec.5.3.1, 35], but considered the possibility of a malicious vote collector (VC) deliberately delaying it. Although the authors of this paper are not very familiar with the technicalities of the down network layers involved in implementing this informing mechanism, it is feasible to implement[25]; IVXV already acquires knowledge of the IP address to calculate ratio of abroad voters as stated in [4].

Finally, all those solutions could be easily made post-quantum by using post-quantum hash based digital signatures, like *Stateless Hash* (*SLH)-DSA* approved NIST standard (see section 6.6), since its relatively large size of 7k Byte is only transmitted once with the download and won't be noticeable by users [57/min 64].

6.3 Range Proofs

To eliminate any complaints (like [14]) and/or risks (ex. [43]) accompanying invalid votes, it is much safer to prevent them from reaching the Ballot processor at all. The thesis in [17] suggested the use of a Zero Knowledge Proof (***ZKP***) check by the Vote collector application to check the validity of the vote it receives from the voting application without revealing it and hence reject invalid votes before being added to the list of votes.
-The thesis preferred the use of ***Range Proofs*** as opposed to ***Set-Membership proofs*** for their simplicity and suggested some mitigations to the discontinuity of the set of vote choices. The authors proposed Range Proofs that are based on ***Bullet Proofs & Pederson Commitments***, since they depend on the Discrete Logarithm problem like

[24] IVXV argues that being able to vote with your application contributes as a measure to the system transparency; our solutions support this feature.

[25] The last lines in (https://stackoverflow.com/questions/35301392/how-to-access-a-remote-desktop-from-a-virtual-machine-set-on-a-server) show that similar things have been done, and (https://serverfault.com/questions/229216/application-which-can-pop-up-like-gtalk-when-some-one-accesses-my-server). Also, any other service in the e-government that links electronic IDs to cellular numbers/emails can receive just the ID to send a fixed message "*Beware you are not using the official voting application*".

El-Gamal encryption used in IVXV. They considered general purpose SNARKs (Succinct Non-Interactive ARgument of Knowledge) based on polynomial commitments not suitable for El-Gamal based voting systems.
-The original paper for Microsoft *ElectionGuard*, [58], also assumed the existence of NIZK (Non-Interactive Zero Knowledge) Range Proof for ballot correctness in their system; the implementation of *ElectionGuard* v.2 used ***disjunctive Chaum-Pederson*** Range Proof[26].
-We could add that whatever the underlying cryptography is, ZKPs that are based on the discrete logarithm problem (like *bullet proofs* and *Chaum-Pederson*) enjoy faster prover time than general purpose SNARKs. This is a desirable quality for ballot validity proof since the voting application is the prover (the vote collector is the verifier); the prover code must be executed online during the voting session and cannot be batched or pipelined even if the used ZKP allows it.
-However, a recent paper, [59], (a follow up to *Kryvos* referred to by [17] in the subject) introduced benchmarks for an efficient implementation of *Groth16* over El-Gamal based voting systems. The authors used a 254-bit common elliptic curve BN254 in their implementation, while IVXV uses 384-bit curve; note that the circuit specific setup needed in Groth16 could be considered a drawback.
-Another 2024 paper, [60], introduces *Polymath* proving it can be more efficient than Groth16 and relates it to KZG commitments[27]. Here, the authors did compare KZG and Groth16 over a 381-bit curve (BLS12-381) which is closer to the one used in IVXV; their work also included mathematical proofs for batching reductions.
Finally, all of this will have to be rethought-of seeking post-quantum validity proofs. There are many post-quantum NIZKs and SNARKs, but the choice must be compatible with the new post-quantum encryption that will be used.

6.4 Insiders' Risk (The Number of Ballots)

We mentioned in section 5.2 that a solution was already implemented to overcome insiders' risks at the processing stage. However, the integrity of the input file still depends on the vote collector and registration application not to collude, which is also an insiders' risk; for example, the Count values are not cryptographically proved. For this vulnerability, we suggest two alternatives; the first depends on checking the consistency of different data sources available in the Estonian government.

Overall Checks

This solution aligns with IVXV solution discussed in section 5.2, since the ***PKIX*** timestamp protocol is used to both record the time of casting the vote and to register the electronic vote in an external independent service. So, instead of trusting the initial *Count* value coming from the vote collector and the registration service, extend to check it with other data records available in the Estonian e-government.

---

[26] https://github.com/microsoft/electionguard-rust/blob/main/TODO.txt#L1373

[27] A condensed summary and a comparison between SNARKs are in the first 25 mins of (https://youtu.be/A3edAQDPnDY); KZG is a name abbreviation.

-A possible general *double check* for the whole list of votes is to compare with the transaction records of the Estonian Information System. In fact, there is an existing *myID* service, [47], that provides what could be viewed as an individual verifiability double check for voters using the *eID* digital identity[28]. We suggest that IVXV performs similar universal verifiability checks on all votes; i.e., checking that *the total number of transactions to IVXV services equals the total number of existing ballots*. Recalling Eq(1), we add the check:

Verify:
Count (original votes file) =
Count_Transactions (source=all, destination=IVXV, time=election_interval)

-If there exist aggregating queries on the information system, the same above check should be repeated for checking the integrity (not just the count) of all votes. The verification process could be a simple hash cascade, a sophisticated ZKP, or even a comparison between sorted versions of the common fields between the two lists:

Verify:
(original votes file) =
Transactions (source=all, destination=IVXV, time=election_interval)

-If aggregating queries were not possible, the first check could be accompanied by some kind of ***Risk Limiting Audits (RLA)***s, where only a sample of random votes could be selected to check manually. RLAs are usually used in e-voting systems that deploy dual paper ballots, [61,62]; here, transactions stored in the Estonian information system could play the role of paper ballots to compare with IVXV data.

```
for all i ∈ sample
{  n= Count (original votes file, vote_ID=i);
   Verify:
   n = Count_Transactions (source=i, destination=IVXV, time=election_interval);
   for all j = 1 to n            //in time order
   Verify:
  Original votes file(vote_ID=i, order=j)=Transaction((source=i, destination=IVXV);
}
```

6.5 The proposed Verkle Tree

A more robust protection from insiders' risk is to cryptographically commit to each vote spontaneously using SNARKs. This choice can be also favorable if the Estonian government does not prefer the interaction between the e-voting system and other e-government entities and yet cares to cryptographically prove the counter values (for each voter and for the whole number of votes) in addition to data integrity.

We suggest ***aggregating vote hashes in an Authenticated Data Structure ADS*** [63/sec.2.1 def.3] which we can simply describe as data structures that can provide succinct (short) cryptographic proof of each element stored in them; we seek one ***that can cryptographically prove the number of values stored in it.*** This way, even if the

[28] (https://x.com/trtram/status/1763936733027049606) demonstrates how a sophisticated user can do that.

VC and RS colluded together, they cannot perform ballot stuffing and/or dropping. In fact, [43] has earlier suggested adding a *NIZKP of knowledge* to each encrypted vote as a protection from privacy attacks. The newer version of IVXV partially responded; the Registration Service now sends *hash(vote)* to the Ballot Processor (Figs. 1&3). Tracing the code [45/line 239], vote hashes are stored in a *Treebag* which represents a binary search tree data structure in Java; i.e., even if it reflected a Merkle tree design, the number of nodes in a Merkle tree (votes) is not cryptographically verified.

In this paper, we suggest the use of ***Verkle Trees*** [64], which is a vector data structure that authenticates its elements based on KZG polynomial commitments (as the polynomial coefficients, and their number reflects the polynomial degree) because they provide cryptographic proof of the number of elements stored in them which is the number of votes in our case. Also, the data structure could be virtual, only the needed proofs have to be kept in storage after runtime. The longer prover time for general purpose SNARKs, mentioned in section 6.3, is not as problematic here, since proofs can be constructed in the background and/or batched using the homomorphic property. Hence, we propose to aggregate all votes in a Verkle Tree (VT); every used Verkle Tree will add a line or 2 to the code {***VT=VT+H[i] * committed_vote***; *i*++;}, where the vector H is calculated in the setup phase.

-If this was used in conjunction with the solution in section 5.2, then we are done here; the Ballot Processor can verify the VT value in its first block (Fig.3), and RLA samples can be verified similarly. If the VT value were published instantaneously when the voting is closed, and the original list of all votes hashes is also available, even independent anonymous verifiers can verify it too. It is also important to make the code publicly available to guarantee the integrity of the VTs construction steps.

-If this solution is to be used to further validate the removal of multiple votes, then it must construct another dynamic final votes VT_2 in parallel; votes from the same voter could be moved to aggregate in second level VTs if needed or accumulated all into a third VT_3 otherwise. There could be a variety of ways to implement this idea depending on the proofs needed; in Fig.4 only step1 is needed in conjunction with [45]. Appendix A contains more details; it also shows an AI generated simulation as a demonstration.

Discussion

Authenticated Data Structures (ADSs) are proposed here as a mean to remove trust assumptions of voting modules by publishing SNARK values that were dynamically calculated using a publicly available code. In this paper, we chose Verkle Tree commitments as opposed to Merkle Trees or STARKs based on the following logic.

-Verkle Trees provide a constant order complexity *SNARK* (per node or per batch that could be all the data) using KZG vector polynomial commitments. Although Verkle Trees require a trusted setup procedure to generate a crs (*common reference string)*, but we do not consider this a problem since it is a universal setup (not circuit specific as in Groth16) which aligns with IVXV setup & key generation phase. This arrangement also allows the use of *Risk Limiting Audits (RLAs*) to verify the details of a selected sample of ballots.

-Traditional Merkle Trees can support checking RLA samples too (the logarithmic complexity is not an issue here since the checks are done offline). However, Merkle trees do not verify the total and subtotals numbers which is crucial in our case.
-Although the ***STARK*** (*Scalable Transparent ARguments of Knowledge*) option introduced in [63] is based on FRI commitments and hence provides post-quantum security guarantee and does not require a trusted setup phase, we find those advantages useless in our case; IVXV involves a key setup phase anyway, and deploys EL-Gamal encryption which is not post-quantum to begin with. In their paper, the authors showed a projected performance analysis for applying their approach to EL-Gamal based e-voting systems, but it is not a performance issue; if quantum computing is feasible, an adversary can discover the private keys and produce correct values that will pass the verifier checks. Hence, we believe STARKs can only be a better solution if they were a suitable match with a post-quantum update of the underlying encryption.

Finally, there is an interesting intuition from [63] that can be useful in archiving IVXV results. Their suggestion to *make the verifier check the generated proof instead of the original data* can be used when election data gets destroyed (after a month); i.e., the VTs generated could be kept forever as permanent proof.

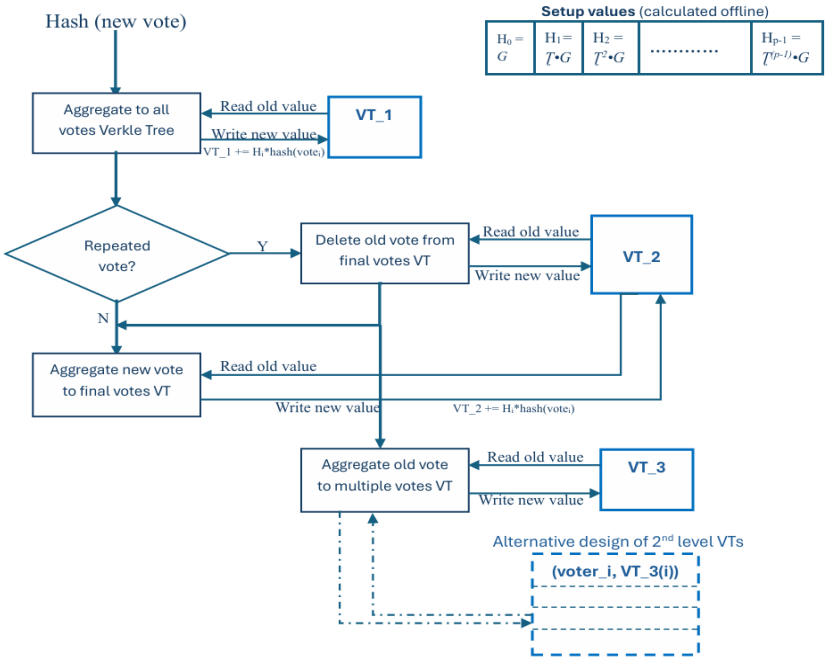

Fig.4. A schematic flowchart of the proposed Verkle Trees design

6.6 Post Quantum Cryptography (PQC)

Quantum computing depends on physics and quantum theory to perform computations much faster than current existing hardware that it could solve all hard problems we currently consider infeasible; specifically, *Shor's Algorithm* [57,66] can solve the Discrete Logarithm and all Elliptic Curve problems, and hence break all cryptographic functions based on their infeasibility. Majority of experts did not believe it is a probable threat [66/Fig.1]; only cryptography researchers discussed it [63,67,68] and started a long and complicated road to post quantum alternatives.

Main Solution Strategies

The *ENISA* (*The European Union Agency for Cybersecurity*) 2021 report, [69], was dedicated to post quantum solutions with a focus on the 3 finalists of NIST round at that time. Suggested solutions [68,69] are based on ***multivariate*** equations over a finite field (ex. UOV and Rainbow), others are based on ***error correcting codes*** (ex. McEliece), but most are based on ***Lattice*** cryptography (ex. Kyber) and/or ***Hash functions***[29] (ex. *SLH-DSA*). ***Lattices***, [70], can be viewed as *n*-dimensional matrices (represent discrete points in an n-dimensional space); this can provide problems that remain hard to compute even with quantum computers like *Shortest Integer Solution (SIS)* and *Learning With Errors (LWE)*. In 2024, [71], the NIST announced 3 PQC standards; a hash based one for digital signatures (*SLH-DSA*), and two Module Lattice-based *ML-DSA* for digital signatures and *ML-KEM* for Key Encapsulation. However, security analysis of *CRYSTALS-Kyber*, the one *ML-KEM* is based on, including vulnerabilities and a *side channel* attack[30] can be found in [72,73].

Threat is Approaching

-Other than research, Quantum computers remained just a theoretical threat that is infeasible to implement; even when believed otherwise, none of the interviewed experts in [74] (May 2024) expects upgrading embedded infrastructure devices to take less than ten years[31]. At the end of last year, things got accelerated rapidly in a way that raised awareness; on Dec 2024, Google & Microsoft unveiled their first chips [75] (although exaggerated by the media [57]); on April 2025 a quantum computing group [76] offered a 1 BTC (i.e. ~100,000$ anticipated cost) reward competition to break a Bitcoin key using quantum computing before April 2026. European financial corporations announced $1bn investments plans on *PQC* in 2025 [77], as opposed to $162.8m value in 2024 [78]; also, NIST latest report anticipates $7bn worldwide [72].

-In general, the retaliation game for political elections is more severe than that of financial systems; some countries are willing to pay much more to break a political election than a thief wanting to steal money. In addition, there is the *Harvest Now Decrypt Later (HNDL)* threat of storing encrypted data to decrypt them later when QC

---

[29] NIST 2016 report, [67], stated that Symmetric key encryption can tolerate by increasing key size; SH-2, SH-3 hash functions by increasing hash length.

[30] Side-channel attacks, [72], discovered by Paul Kocher in 1996 as *Timing attacks* on implementations; they learn information from data disclosed while a device is functioning like electromagnetic radiation, sound waves, power use, or execution time.

[31] [66] puts the first obstacle to PQ transition as: "*The urgency of starting the PQ migration is not well understood by decision-makers*", while [57] warns from unstudied rushing.

is possible [57]. Hence, all countries deploying e-voting/e-government systems should be more motivated, without uncarefully studied rush, to provide post-quantum solutions; in fact, e-governments/e-IDs would be a probable main target of a quantum adversary than e-voting systems as stated by most interviewed experts in [74].

Estonian Quantum Efforts

For our specific case of Estonia, the authors of [74] included 5 out of 24 experts from Estonia in their interviews, and covered steps taken by the Estonian government [74/ sec. 2.2.2] in different quantum related aspects; a European cooperation to deploy a quantum communication infrastructure in 2020 [79] which led to the NordIQuEst (*Nordic-Estonian Quantum Computing e-Infrastructure Quest*) in 2022 [80], a collaboration between *Cybernetica* (IVXV company) and University of Tartu to create post quantum solutions has started in 2021 [81] including jointly supervised doctoral thesis from which we encountered a paper (2024) on PQC migration obstacles in Estonia [66], and finally PQC is 1 of 6 challenge areas included the Cyber Security hub established in 2023 between Estonia and a major Czech ICT powerhouse both in industry and education [82]. Then, the European Commission published on 11 April 2024, [83], a coordinated PQC implementation roadmap for all member countries. The announcement contained only one guiding statement; the implementation should be "*via hybrid schemes that may combine Post-Quantum Cryptography with existing cryptographic approaches or with Quantum Key Distribution (QKD)*". This was different from ENISA 2021 report [69/page 35] which excluded solutions using *QKD* because its setup relies on pre-established authenticated communications channels; [74,84] also reported different experts' opinions on this point. In fact, [66] considered "*EuroQCI focused attention on QKD technology"* as an obstacle to post quantum transition since they believe its functionality is limited compared to PQC.

Estonian I-Voting Related Efforts

-Digital IDs are a crucial element in i-voting; the *ENISA* July 2024 report [85] on the unified European digital identity project *eIDAS*, though, did not specify the PQC transition details. Then on 19 Feb 2025, [86], *Cybernetica* announced that the first hardware chip enabling post-quantum ID card has been certified in Europe; when asked about post quantum i-voting, they answered "*there is more math to do*". In their 2024 paper [66/sec. 2.E], they considered the complexity of managing multiple layers a disadvantage in most existing Hybrid systems. For i-voting, [66/sec. 4.E], this would be a problem with the needed multiple layers to shuffle votes in *mix-nets*; we think lattice-based approaches maybe advantageous here, [84], as they depend on matrix operations. Brief survey on post quantum i-voting research attempts and some post quantum replacements for El-Gamal encryption is in [65].

7 Possible Research Directions

In this section we go through possible research threads that could evolve from the gathered material, even if not necessarily pursued by the authors of this paper.

The work in [59] could be a starting point for researchers concerned with hardware acceleration of Zero Knowledge Proofs that goes in depth into circuit specific details, while [60] would be useful for researchers interested in theoretical cryptographic proofs. Also, research can be further pursued to conduct comparative analysis between possible zero knowledge proofs concerning implementation details along proof batching reductions over the exact 384-bit elliptic curve used in IVXV.

However, one may think the future will promote more research on efficient quantum secure solutions and less encouragement to elliptic curve research. The research space is challenging, [68], on the efficiency-security tradeoff; demanding for new innovations [70], for cryptanalysis [72,73], and formal verification [84] of the new PQ protocols (how to prove correctness and security if it is not verifiable on existing hardware [57]). In addition, AI-based attacks could be another rich research area; side channel attacks, [73], "*how information can be leaked from the implementation of a cryptographic component*" and maybe from quantum devices too.

Finally, with the intuition of [53], it could be an interesting research to tackle the broader question of *to what limit can the information provided by general purpose activity logs of digital identities* (in Estonia or any country that uses digital identities in online voting) *help vote buyers/coercers* in catching voters who try to deceive them, and whether a blockchain based e-government is an advantage or disadvantage in that direction. We have shown in section 6.4 the information *myID* [47] and similar services can provide; to what limit could adversaries expand their toolbox using them to defeat, for example, the *CK* (*Complete Knowledge*) authenticity metric [65/footnote 21] whether for vote buying/coercion or any other malicious activity.

8 Summary & Conclusions

In this paper we gave a political and technological historical brief on the development and status quo of the Estonian internet voting system. Then we explained the current system architecture and surveyed available material from the academic literature and different other resources to cover reported attacks and/or vulnerabilities and how they were fixed by the system.

We addressed the concerns of opposing parties in their meeting with ODHIR [9/sec.3.3/28], and introduced solutions to some vulnerabilities; mainly, authenticating the voting application and cryptographically proving the list of votes against insiders' manipulation.

We suggested some possible solutions that give voters the freedom to deliberately use a different voting application they trust more; the first is allowing the voter to check the application fingerprint prior to voting; the second is allowing only a list of pre-registered applications (through digital signatures) and informing the voter with the result. In addition, we introduced the threat of *encrypted copy attacks* that could

be conducted at large scale, [87,88], using an adversary's voting application; hence, we highly recommend IVXV to adopt a solution to the voting application problem. We also recommend deploying Range-proofs as suggested by Tallinn University thesis, [17], to get rid of invalid votes problems.

Then, we proposed alternatives for more robust guarantees that the list of votes was not manipulated by insiders; one is to extend the currently deployed consistency checks and add RLAs comparing ballots to general records of digital transactions interacting with IVXV. The second is aggregating votes (online during the voting process) in an authenticated data structure like Verkle Trees which we detailed as our proposed solution. This will eliminate the possibility of colluding insiders performing the above attack and will contribute to more public trust.

Finally, we highlighted the quantum computing threat and ended by introducing a variety of possible research threads that could evolve from all the paper material.

As a closure, we advise the IVXV team to respond more scientifically and objectively to OSCE, complaints, and researchers; this will give them more credit for their unnoticed work and will lead to a better understanding.

Acknowledgements

I'm grateful to everyone participated in making me acquire this level of knowledge & research skills: from my graduation faculty and postgraduate supervisors nearly 30 yrs ago, to everyone who has put their work free online (Berkeley ZKP MOOC, HAL, iacr, arXiv, E-Vote-ID, Tallinn University); also, many useful links were reached through talking to people on social media platforms. Finally, this work benefited from AI companions (grok, Gemini, and Capilot) in 2 ways; the first is asserting thoughts as discussing with a colleague researcher as in [56,87], the second is repeatedly generating AI review reports to discover weakness points and prepare a revised version.

References

1. Piret Ehin, Mihkel Solvak, Jan Willemson, and Priit Vinkel, "*Internet voting in Estonia 2005–2019: Evidence from eleven elections*", Oct 2022; https://doi.org/10.1016/j.giq.2022.101718; https://www.sciencedirect.com/science/article/pii/S0740624X2200051X
2. OSCE/ODHIR 2023 report on Estonian Internet Voting, https://osce.org/files/f/documents/f/f/551179_0.pdf
3. https://www.smartmatic.com/featured-case-studies/estonia-the-worlds-longest-standing-most-advanced-internet-voting-solution/; last accessed 30/6/2024.
4. https://www.valimised.ee/en/archive/statistics-about-internet-voting-estonia; last accessed 30/4/2025.
5. https://gafgaf.infoaed.ee/en/posts/great-divide-in-evoting/; last accessed 14/3/2024.
6. https://ausadvalimised.ee/docs/yhisavaldus2023/; and newer petitions in 2024: https://ausadvalimised.ee/ei-lepi-vaadeldamatusega/, last accessed 13/6/2024; https://x.com/ausadvalimised/status/1808854585597108552, last accessed 5/7/2024.

7. Alexander Martin, 13/3/2023, https://therecord.media/estonia-cyberattack-parliamentary-elections, last accessed 8/7/2025.
8. Jan Willemson, "*Recommendations to OSCE/ODIHR (on how to give better recommendations for Internet voting)*", 10 Feb 2025, https://arxiv.org/html/2502.06385v2; last accessed 25/6/2025.
9. OSCE/ODHIR, "*Opinion on the Regulations of Internet Voting in Estonia*", 17 June 2025, https://osce.org/files/f/documents/e/a/593435.pdf
10. "*A computer scientist made available the code for e-elections, which the electoral service has so far been fiercely hiding*", https://digi.geenius.ee/eksklusiiv/arvutiteadlane-tegi-kattesaadavaks-e-valimiste-koodi-mida-valimisteenistus-on-seni-kiivalt-varjanud/; last accessed 2/1/2024.
11. https://gafgaf.infoaed.ee/en/posts/perils-of-electronic-voting/; last accessed 4/1/2024.
12. https://media.ccc.de/v/37c3-12298-should_e-voting_experience_of_estonia_be_copied#t=965; last accessed 15/1/2024.
13. Post Times, "*The use of e-voting should be limited*", https://arvamus.postimees.ee/7974894/mart-poder-e-haaletuse-kasutust-tuleks-piirata; last accessed 13/3/2024.
14. Election Commission of the Republic, "*Resolution of Andres Alla's complaint*", 21.06.2024 No. 14, https://www.riigiteataja.ee/akt/322062024003; last accessed 5/7/2024.
15. National Electoral Commission, "*Resolving Märt Põdra's Complaint*", 20/6/2025, https://www.riigiteataja.ee/akt/321062025005; last accessed 29/5/2025.
16. https://www.valimised.ee/en/internet-voting/observing-auditing-testing; last accessed 5/7/2024.
17. Taaniel Kraavi, "*Proving Vote Correctness in the Estonian Internet Voting System*", Master thesis, Tallinn University of Technology, June 2024, https://digikogu.taltech.ee/et/Download/ffdf0de1e58d455ba3d484400c9123fc.pdf
18. "*A transparent digital ballot box can be tried in the e-voting threshold survey*", https://ausadvalimised.ee/uuenduslik-exitpoll/; https://github.com/infoaed/pseudovote-euro24/tree/JUNE5TH2024; last accessed 5/7/2024.
19. Ago Samoson, "*The developers of our e-election system could admit their strategic mistake in order to prevent the worst*", 17/3/2024; https://arvamus.postimees.ee/7981474/ago-samoson-valimishavingut-tuleb-ennetada, last accessed 9/7/2024; on 17/3/2025, https://arvamus.postimees.ee/8211830/ago-samoson-valimised-ehk-e-mang-koduvaljakul,
20. "*E-voting system Disk appeared out of nowhere*", https://gafgaf.infoaed.ee/posts/esoteeriline-turvamudel/; last accessed 22/5/2024.
21. Election Commission of the Republic, "*Review of Mart Podra's Complaint*", 23/2/2023, https://www.riigiteataja.ee/akt/328022023004; last accessed 7/7/2024.
22. The report of the cyber security committee of the Academy of Sciences, https://x.com/danbogdanov/status/1802998209649762582; last accessed 6/7/2024.
23. Cyber Security Commission minutes of meetings, https://www.akadeemia.ee/akadeemia/noukogud-ja-komisjonid/kuberturvalisuse-komisjon/; last accessed 7/7/2024.
24. https://ec.europa.eu/digital-building-blocks/wikis/pages/viewpage.action?pageId=533365949; last accessed 28/12/2023, https://scoop4c.eu/cases/estonian-internet-voting; last accessed 22/11/2023.
25. Arnis Parsovs, "*Estonian Electronic Identity Card: Security Flaws in Key Management*", 29th USENIX Security Symposium, Aug 2020, 978-1-939133-17-5; video

https://www.usenix.org/conference/usenixsecurity20/presentation/parsovs; last accessed 24/3/2024.

26. Arnis Parsovs, "*Estonian Electronic Identity Card and its Security Challenges*", 2021, PhD Thesis, University of Tartu.
27. Geenius, *"The police discovered 15,000 faulty ID cards, over 300 have been used (in Estonian)*", June 2019. https://digi.geenius.ee/rubriik/uudis/politsei-avastas-15-000-veaga-id-kaartiule-300-on-kasutatud/; last accessed 20/3/2024.
28. Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, Vashek Matyas, "*The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli*", CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pages1631 - 1648, https://dl.acm.org/doi/10.1145/3133956.3133969
29. https://e-estonia.com/raulwalter-estonia-digital-identity-giant/; last accessed 20/3/2024
30. https://e-estonia.com/estonia-introduced-a-new-id-card/; last accessed 20/3/2024.
31. https://valimised.ee/sites/default/files/2023-02/IVXV-protocols.pdf
32. Kristjan Krips, "*Privacy and Coercion Resistance in Voting*", June 2022, PhD Thesis, University of Tartu, dspace.ut.ee/server/api/core/bitstreams/58ffcbf3-7cc8-4381-b7ca-a9d3e777dcd6/content; last accessed 10/7/2025
33. Jan Willemson, "*Creating a Decryption Proof Verifier for the Estonian Internet Voting System*", ARES 2023, Italy, ACM ISBN 979-8-4007-0772-8/23/08, https://doi.org/10.1145/3600160.3605467
34. Anggrio Sutopo, Thomas Haines, Peter Rønne. "*On the Auditability of the Estonian IVXV System and an Attack on Individual Verifiability*". Workshop on Advances in Secure Electronic Voting, May 2023, Bol, brac, Croatia. hal-04216242; https://halscience/hal-04216242
35. Olivier Pereira, "*Individual Verifiability and Revoting in the Estonian Internet Voting System*", 2022, https://www.researchgate.net/publication/372570425_Individual_Verifiability_and_Revoting_in_the_Estonian_Internet_Voting_System
36. S. Heiberg and J. Willemson, "*Verifiable Internet Voting in Estonia*", 2014, 6th International Conference on Electronic Voting: Verifying the Vote (EVOTE), Austria, pages 1-8, https://ieeexplore.ieee.org/document/7001135
37. D. Springall et al., "*Security Analysis of the Estonian Internet Voting System*", In Proceedings of the 2014 ACM SIGSAC Confrence on Computer and Communications Security (CCS'14), pages 703-715, https://dl.acm.org/doi/10.1145/2660267.2660315
38. https://research.cyber.ee/~janwil/publ/ivxv-evoteid.pdf
39. Smartmatic-Cybernetica. IVXV Voting Service. Version 1.8.2-RK2023, https://github.com/valimised/ivxv/tree/master
40. https://github.com/valimised/ivotingverification/blob/published/app/src/main/java/ee/vvk/ivotingverification/util/ElGamalPub.java#L77-L83, and https://github.com/valimised/ios-ivotingverification/blob/published/VVK/Crypto.m#L141-L146
41. Valimised, "*Protocols PDF"*: https://www.valimised.ee/sites/default/files/2024-05/RVT%20korraldus%20nr%2012%20lisa%20%28IVXV-protokollide%20kirjeldus%29.pdf; "*Architecture PDF*": https://www.valimised.ee/sites/default/files/2024-05/RVT%20korraldus%20nr%2012%20lisa%20%28IVXV-arhitektuur%29.pdf
42. The *Decrypt tool in the key application*, IVXV 1.9.10 EP2024, https://github.com/valimised/ivxv/blob/published/key/src/main/java/ee/ivxv/key/tool/DecryptTool.java#L247; last accessed 5/7/2025

43. J. Müller, "*Breaking and Fixing Vote Privacy of the Estonian E-Voting Protocol IVXV*", In: Matsuo, S., et al. Financial Cryptography and Data Security. FC 2022 International Workshops. FC 2022. LNCS(13412), https://doi.org/10.1007/978-3-031-32415-4_22
44. Krips, K., Snetkov, N., Vakarjuk, J., Willemson, *"Trust Assumptions in Voting Systems"*. In Computer Security. ESORICS 2023 International Workshops. Lecture Notes in Computer Science, vol 14399. Springer, Cham, https://doi.org/10.1007/978-3-031-54129-2_18; full paper at https://arxiv.org/pdf/2309.10391
45. https://github.com/valimised/ivxv/blob/published/auditor/src/main/java/ee/ivxv/audit/tools/IntegrityTool.java
46. Tarvo Treier and Kristjan Duuna, "*Identifying and Solving a Vulnerability in the Estonian Internet Voting Process: Subverting Ballot Integrity Without Detection*", IEEE Access, Vol.12, https://ieeexplore.ieee.org/document/10811882
47. https://myid.skidsolutions.eu/en; last accessed 13/7/2024.
48. Sevdenur Baloglu, Sergiu Bursuc, Sjouke Mauw, and Jun Pang, "*Formal Verification and Solutions for Estonian E-Voting*", In ACM Asia Conference on Computer and Communications Security (ASIA CCS '24), July 2024, Singapore, Singapore. ACM, New York, NY, USA, https://doi.org/ 10.1145/3634737.3657009
49. Interaction with IVXV, 3/2024, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/Letter_to_IVXV_with_replies.pdf
50. https://github.com/valimised/ivxv/blob/published/voting/internal/sessionstatus/rpc/client.go(#L22,#L103)
51. Vincent Cheval, Charlie Jacomme, Steve Kremer, and Robert Künnemann, "*SAPIC+: Protocol Verifiers of the World, Unite!*", In 31st USENIX Secu rity Symposium, USENIX Security 2022, Boston, MA, USA, August 2022, Kevin R. B. Butler and Kurt Thomas (Eds.). USENIX Association, 3935–3952, https://www.usenix.org/conference/usenixsecurity22/presentation/cheval
52. https://www.valimised.ee/en/internet-voting/documents-about-internet-voting
53. PMPhilip Daian, Tyler Kell, Ian Miers, and Ari Juels; July 2018; https://hackingdistributed.com/2018/07/02/on-chain-vote-buying/
54. James Austgen, Andrés Fábrega, Sarah Allen, Kushal Babel, Mahimna Kelkar, Ari Juels, "*DAO Decentralization: Voting-Bloc Entropy, Bribery, and Dark DAOs*"; https://arxiv.org/abs/2311.03530; https://github.com/DAO-Decentralization/dark-dao/tree/main; last accessed 20/3/2024.
55. https://bitcoinelectrum.com/how-to-verify-your-electrum-download/; last accessed 17/3/2025.
56. Grok, "*Batching Electrum-like Checks in 1-Button Click for Authenticating IVXV Voting Application*", 9/7/2025, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/Grok_X_Electrum_1button_IVXV.pdf
57. https://a16zcrypto.com/posts/podcast/quantum-computing-what-when-where-how-facts-vs-fiction/; last accessed 17/5/2025.
58. J. Benaloh, M. Naehrig, O. Pereira, and D. S. Wallach, "*ElectionGuard: a Cryptographic Toolkit to Enable Verifiable Elections*", https://eprint.iacr.org/2024/955, https://www.electionguard.vote/spec/; last accessed 14/7/2024.
59. N. Huber et al, "*ZK-SNARKs for Ballot Validity: A Feasibility Study*", E-Vote-ID 2024, pp 107-123, Oct 2024; https://link.springer.com/chapter/10.1007/978-3-031-72244-8_7
60. H. Limpaa, "*Polymath: Groth16 Is Not The Limit*", CRYPTO 2024, pp 170-206, Jun 2024; https://link.springer.com/chapter/10.1007/978-3-031-68403-6_6

61. Risk Limiting Audits (RLAs), "*What is RLAs*", https://verifiedvoting.org/audits/whatisrla/; last accessed 20/5/2025.
62. "*RLAs Frequently Asked Questions*" https//www.sos.state.co.us/pubs/elections/RLA/faqs.html; last accessed 20/5/2025.
63. Max Harrison and Thomas Haines, "*On the Applicability of STARKs to Counted-as-Collected Verification in Existing Homomorphically E-Voting Systems*", Mar 2024; https://fc24.ifca.ai/voting/papers/Voting24_HH_On_the_Applicability_of_STARKs_to_Counted-as-Collected_Verification_in_Exisitng_Homomorphically_E-Voting_Systems.pdf
64. Zero Knowledge Berkely MOOC 2023, lecture 5, "*KZG polynomial commitment scheme"*; https://youtu.be/tAdLHQVWlUY; last accessed 13/7/2024.
65. Extended version of the paper, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/Estonia_EVoteID_long.pdf
66. J. Vakarjuk, N. Snetkov, and P. Laud, "*Identifying Obstacles of PQC Migration in E-Estonia*", 2024, https://ieeexplore.ieee.org/document/10685570; full paper available at https://ccdcoe.org/uploads/2024/05/CyCon_2024_Vakarjuk_Snetkov_Laud-1.pdf
67. Bello A. Buhari, Afolayan A. Obiniyi, Sahalu Junaidu, and Abubakar Roko, "*Elgamal Cryptographic Scheme based on Quantum Key Distribution (QKD)*", Dec 2020, DOI: 10.47310/iarjet.2020.v01i04.008, https://www.researchgate.net/publication/344784508_Elgamal_Cryptographic_Scheme_based_on_Quantum_Key_Distribution_QKD
68. T. Niraula et al., "*Quantum Computers' threat on Current Cryptographic Measures and Possible Solutions*", Oct 2022, International Journal of Wireless and Microwave Technologies 12(5):10-20, DOI:10.5815/ijwmt.2022.05.02, https://www.researchgate.net/publication/368394434_Quantum_Computers'_threat_on_Current_Cryptographic_Measures_and_Possible_Solutions
69. https://www.enisa.europa.eu/publications/post-quantum-cryptography-current-state-and-quantum-mitigation; last accessed 27/4/2025.
70. Dana Sairangazhykyzy Amirkhanova, Maksim Iavich, and Orken Mamyrbayev, "*Lattice-Based Post-Quantum Public Key Encryption Scheme Using ElGamal's Principles*", Cryptography 2024, 8(3), 31; https://doi.org/10.3390/cryptography8030031
71. https://quantumcomputingreport.com/nist-has-finalized-the-first-three-pqc-algorithms-45-more-are-still-in-the-pipeline/#; last accessed 18/5/2025.
72. Iavich, M. and Kuchukhidze, T., "*Investigating CRYSTALS-Kyber Vulnerabilities: Attack Analysis and Mitigation*", Cryptography 2024; https://www.mdpi.com/2410-387X/8/2/15
73. Grünfeld and J. Mathias H., "*Side-Channel Attacks on CRYSTALS Kyber: An Analysis of a Post-Quantum Algorithm and Its Vulnerabilities to Side-channel Attacks*", Master's Thesis, NTNU, Trondheim, Norway, 2023.
74. A.R. Perez et al., "*An electoral exception? Quantum computing - readiness and internet voting*", JeDEM Issue 16 (3): 50-79, 2024, DOI: 10.29379/jedem.v16i3.928
75. Chris Vallance, "*Google unveils 'mind-boggling' quantum computing chip*", https://www.bbc.com/news/articles/c791ng0zvl3o, last accessed 22/4/2025; Google published paper: https://www.nature.com/articles/s41586-024-08449-y
76. https://www.coindesk.com/tech/2025/04/17/quantum-computing-group-offers-1-btc-to-whoever-breaks-bitcoin-s-cryptographic-key; last accessed 22/4/2025.
77. https://digital-strategy.ec.europa.eu/en/news/commission-publishes-recommendation-post-quantum-cryptography; last accessed 22/4/2025.
78. https://uk.finance.yahoo.com/news/europe-post-quantum-cryptography-market-151700764.html; last accessed 23/4/2025.

79. https://www.mkm.ee/en/news/estonia-joined-eus-cooperation-framework-quantum-communication-infrastructure; last accessed 28/4/2025.
80. https://neic.no/news/2022/08/19/nordiquest-has-started/; last accessed 28/4/2025.
81. https://sciencebusiness.net/network-updates/university-tartu-and-cybernetica-cooperate-study-quantum-safe-cryptography; last accessed 28/4/2025.
82. https://cordis.europa.eu/project/id/101087529; last accessed 28/4/2025.
83. https://digital-strategy.ec.europa.eu/en/library/recommendation-coordinated-implementation-roadmap-transition-post-quantum-cryptography; last accessed 27/4/2025.
84. ZK podcast, "*Lattice based ZK Systems with Vadim Lyubashevsky*", Episode 359, 30th April 2025, https://youtu.be/yQD65PhFwwY; last accessed 1/5/2025.
85. https://www.enisa.europa.eu/sites/default/files/2024-11/Remote%20ID%20Proofing%20Good%20Practices_en_0.pdf; last accessed 22/4/2025.
86. https://e-estonia.com/cybernetica-post-quantum-cryptography-joins-to-menu/; last accessed 28/4/2025.
87. Grok, "*Assessing the paper statements about the Encrypted Copy Attack*", 7/7/2025, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/ encrypted_copy_attack_grok.pdf
88. Gemini, "*Assessing the paper statements about the Encrypted Copy Attack*",12/7/2025, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/Google_Gemini_encrypted_attack.pdf
89. Gemini, "*Simple Simulation for Vote Proofs (using hashes not real KZG)"*, 13/7/2025, https://g.co/gemini/share/9c2515e07774; https://g.co/gemini/share/cc86b8d281dd
90. Copilot, "*Python Simulation for all Verkle Tree Steps*". 13/7/2025, https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/ Copilot_Verkle_complete_Chat.pdf; wrapped code: https://anonymous.4open.science/r/SoK_Estonia_IVXV_EVoteID-C2E4/verkle_cli.py
91. Jan Oberst, "*Towards Stateless Clients in Ethereum: Benchmarking Verkle /trees and Binary Merkle Trees with SNARKs*", 18 April 2025, https://arxiv.org/html/2504.14069v1
92. https://www.wisekey.com/press/wisekey-pki-and-sealsq-post-quantum-technologies-enhance-e-voting-security-through-advanced-cybersecurity-and-ai-integration/
93. I. Chillotti, N.s Gama, M. Georgieva, and M. Izabach`ene. "*A homomorphic LWE based e-voting scheme*", 2016, In PQCrypto, volume 9606 of LNCS, pages 245–265. Springer
94. Rafa¨el del Pino, Vadim Lyubashevsky, Gregory Neven, and Gregor Seiler, "*Practical quantum-safe voting from lattices*", 2017, In ACM Conference on Computer and Communications Security, pages 1565–1581. ACM.
95. Guillaume Kaim, Sébastien Canard, Adeline Roux-Langlois, Jacques Traoré, "*Post-quantum Online Voting Scheme*", FC 2021 - Financial Cryptography and Data Security. International Workshops, Mar 2021, Virtual event, France. pp.290-305, DOI:10.1007/978-3-662-63958-0_25 ;https://hal.science/hal-03355875
96. https://www.researchgate.net/publication/360161016_A_Review_of_Cryptographic_Electronic_Voting
97. S. Shriya, J. D. Sweetlin and V. Supraja, *"Secure Online Voting System Using Hybrid Post-Quantum Signatures,"* 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2024, pp. 1-7, doi: 10.1109/ACCAI61061.2024.10602388.
98. Hough, Sandsbråten, and Silde , "*More Efficient Lattice-Based Electronic Voting*", 13-Jan-2025, from NTRUIACR Communications in Cryptology10.62056/a69qudhdj**1**:4; https://cic.iacr.org/p/1/4/10/pdf

99. Bello A. Buhari, Afolayan A. Obiniyi, Sahalu Junaidu, and A.F.D. Kana, "*Enhancement of S13 Quantum Key Distribution Protocol by Employing Polarization Secrete Key Disclosure and Non-repudiation*", Aug 2023, International Journal of Wireless and Microwave Technologies, https://www.researchgate.net/publication/372769399_Enhancement_of_S13_Quantum_Key_Distribution_Protocol_by_Employing_Polarization_Secrete_Key_Disclosure_and_Non-repudiation

Appendix A: The detailed steps of a complete Verkle Tree (VT) Solution

A.1 Setup:
The election authority picks a random secret τ to be destroyed after the end of the setup procedure and calculates (offline) the following terms with a finite cyclic group G that is closed over the used finite field [64]; the calculated terms should be public to verifiers:
$H_0 = G$, $H_1 = \tau \bullet G$, $H_2 = \tau^2 \bullet G$, ..., $H_{p-1} = \tau^{(p-1)} \bullet G$

A.2 Steps:
-When a new vote enters the system (through the vote collector and the registration applications), it goes through the following steps (Fig.6):

1. When a new vote record enters the vote collector, in addition to adding it to the votes list, the system cryptographically commits to the vote record by adding its hash to the first Verkle Tree; VT_1 = VT_1 + Hash(vote_i)*H_i
2. If this voter-ID hasn't appeared before, the new vote hash is aggregated into the second VT as well; VT_2 = VT_2 + Hash(vote_i)*H_i
3. If it's a repeated voter-ID, then before aggregating it to the second VT, the previous vote must be deleted from it first[32]. The deleted vote is then aggregated into the third VT (or the corresponding second le\vel VT if needed). Since [38] shows that multiple voters are sparse, O(10,000) multiple votes, second level VTs could be implemented as an array holding (voter-ID, VT value)
4. The fact that a voter has voted at the polling station, will not be available during this phase to separate them; if Range proofs were not deployed, then the separation of invalid votes will not be possible at this phase too. However, the Ballot Processor can commit to their values (after separating them as in Fig.6) by calculating VT_4 and VT_5 respectively.

-When i-voting time is closed, make all VT values publicly available in an immutable way; this could be done through the official election site or any trusted robust blockchain.

[32] We assume the index of the old multiple vote make it retrievable from the list of votes, otherwise step3 will be done by the Ballot Processor; also, this could be done in the background without delaying the interactive processing with voters.

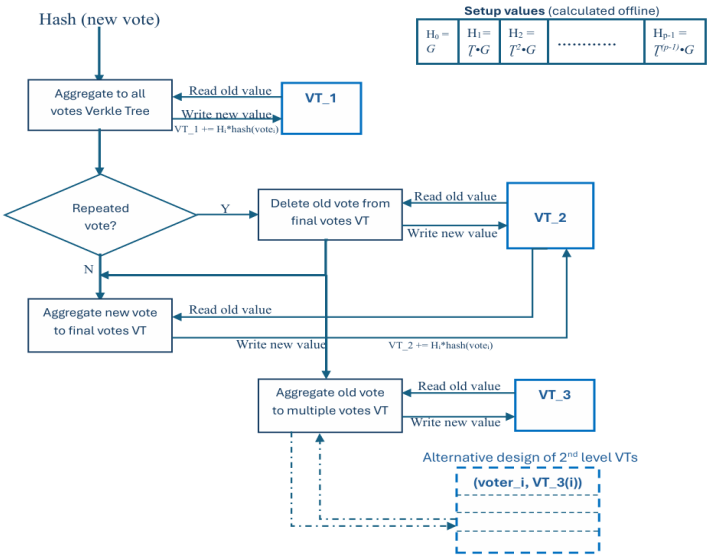

Fig.4. A schematic flowchart of the proposed Verkle Trees design

A.3 Results

The proposed Verkle Trees adds the following security values:

- The polynomial degree of each Verkle Tree proves the number of votes it holds (*n* of a Verkle Tree is cryptographically proved); i.e., two values from Eq.1 in section 5.2: *Count (original votes file)* and *Count (multiple votes).* The Ballot Processor should check that ***n(VT_1)= n(VT_2) + n(VT_3)***[33].
- In addition to the above check, publishing all the VT commitments allows independent verifiers to write their own code to calculate VT_end of anonymized votes. Then, verify ***n(VT_end) = n(VT_2) – n(VT_4 + VT_5)***.
- If RLAs were applied and samples were selected to verify manually, the calculated VT values can also provide separate succinct proof for every sampled value; this is possible whether samples were taken as ballots or as voters. The alternative design of 2nd level VT_3 (Fig.7) may facilitate individual voter verification in a more detailed way; VT_3(i) can prove the number and values of every deleted (multiple) vote for the sampled voters, even if they voted at a poll station afterwards.
- In the 2-levels VT_3 case, **QR codes can include the number of multiple votes** for each voter; this is an extra check against vote manipulation[34].

[33] There could be a variety of implementations that could prove that the values in the sub-vectors combines to exactly the values in the all entered votes vectors; basically, the values (ballots/ballots hashes) are treated as the polynomial coefficients and one can work around that.

[34] There is a maximum of 3 QR code checks, but a voter could vote 10 times and check only the QR of the last vote; in this case our modified QR will reply that "*you voted 10 times, your last vote is…*".

- **VT_**s could be kept after destroying election data to verify individual votes.

A.4 Simulation

-We asked Gemini, Google AI companion, to write a demonstrative simulation tool to show how this could be done within voting [89]; it prepared 2 versions that use simplified functions (not real KZG commitments) to demonstrate the steps. Fig.5 shows a sample run.

-Microsoft copilot, [90], on the other hand completed the whole steps using some open source from Ethereum Verkle Trees implementation; it used Elliptic Curve BLS12-381 with real KZG math (via py_ecc). We are certain that if IVXV team decided to adopt this solution, they will write their own code; the point here is to show it is feasible to build and batch in the background to not delay the voting process.

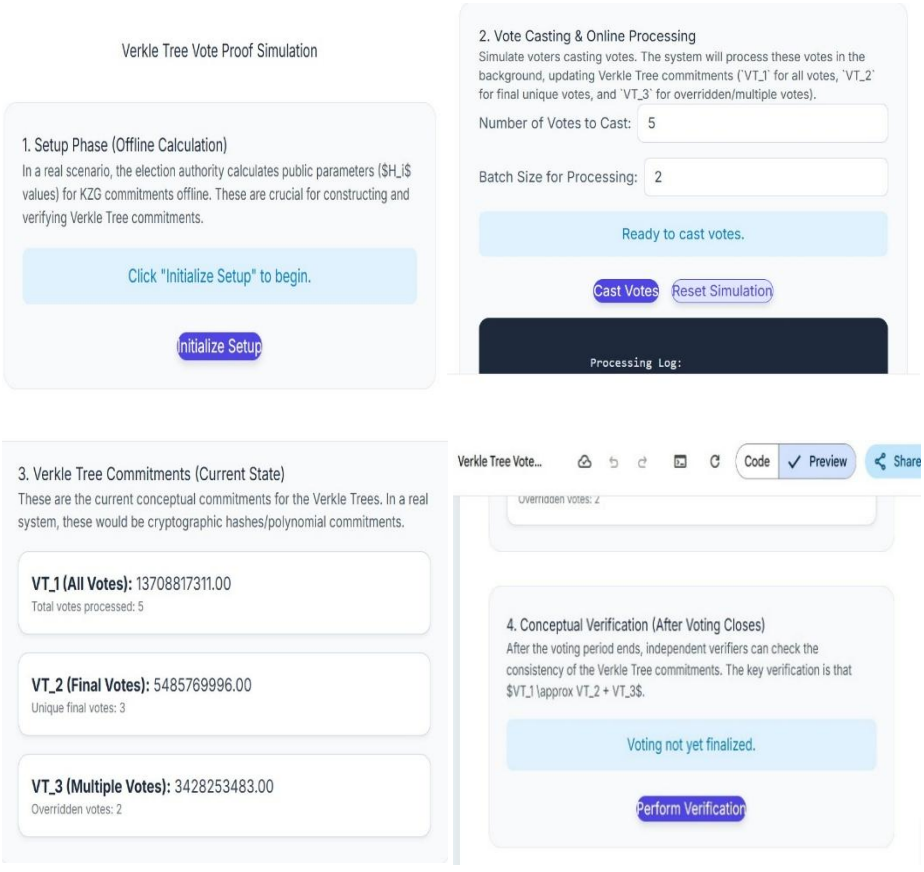

Fig.5: Screenshots from a sample run of [89]

Note that Gemini simulation is just a demonstration; cryptographic KZG coefficients are not calculated. Hence, the relation between the VT values is not simple addition as Gemini wrote[33].

A.5 Impact on Efficiency

It is worth mentioning that Verkle Trees are used by the Ethereum blockchain where a block of transactions is finalized every 10 seconds on average; i.e., no long delays are expected.

This result is further confirmed by this recent paper [91] that compared benchmark results for Verkle Trees (implementation from [91/ref.19]) versus Merkle Trees and an accompanying SNARK (Plonk [91/ref.6]); the paper found Verkle Trees to be faster and more efficient in proof generation, while the Merkle solution has a faster verification time. ***This exactly matches the needs of an online i-voting system where proof generation happens online, while verification happens offline after voting is closed***. The exact curves can be found in [91/sec V] where for regular Windows 10 22H2 operating system on an Intel i5-4690K machine with 22 GB ram, the proof size remains below 2 seconds as long as number of nodes $< 2^{22} \sim 4.19$ million.

We believe this is a relieving upper bound, since the upper bound on the number of electronic votes is less than half this number and the calculating server machine will sure have more RAM; for an AMD Ryzen 32-core and 125 GB the proof generation is nearly 1 second and remains less than 2 even when the number of nodes reaches $2^{25} \sim$ 33 million.

Appendix B: A Brief on Post Quantum Online Voting and Replacements of El-Gamal Encryption

1 Companies

We have discussed Cybernetica in the main text; [86] explains that they have different countries as customers some of which have 2030 PCQ migration plans, while Estonia is expected to be sooner than that. Another software company, Wisekey located in Switzerland [92], offers its post quantum e-voting services with AI and blockchains as extra lines of defense; however, we did not encounter any real deployment yet.

2 PCQ i-voting

There are few earlier attempts, all of which are lattice-based; [93/sections 4.5&6] presented a comparative summary and found, as of 2022, post quantum e-voting research to be *in its initial stages and has not been fully developed*. Also Cybernetica-Tartu paper in 2024 [66] identified "*research lacking on PQC for esoteric use cases such as i-voting*" as one of the obstacles.

Exploring few existing papers, we find [94] from 2016 relied on the honesty of the Bulletin Board and was inefficient for depending on fully homomorphic encryption without ZKPs, then in 2017 [95] deployed ZKPs and proof batching to be more efficient but still the number of needed proofs increases logarithmically with the number of candidates, while [96] (2021) overweighs the election authority trust assumption which is highly unrecommended in the Estonian case. Another 2024 paper [97] that applies Lamport signatures on Merkle data (to decrease the data size); although seems promising, the Merkle tree hash-based signature grows with the number of leaves linearly in the private key size and logarithmically in the signature size (with large constant as well, 256*log n)[35]. Finally, we mention [98] from 2025 which is a lattice-based e-voting protocol following the standard mix-and-decrypt framework and supports general ballots.

2 PCQ El-Gamal replacements

Although dated in 2020, [67] presented a straightforward solution For El-Gamal based systems where the user(voter)'s public key is encrypted by a symmetric encryption algorithm (Blowfish in [67] but could be AES for more security) and then shared using *QKD* (*Quantum Key Distribution*); the same quantum key is used for the symmetric encryption. In this arrangement the resulting public key can only be used once, which may complicate multiple voting; the future work of [67] included the use of *S13 QKD* which can perform the encryption and public key sharing in one step, then they introduced further enhancement, [99] on S13 in 2023.

Another newer (2024) alternative for El-Gamal replacement that aligns better with ENISA recommendations and NIST final standards is introduced in [70]; a lattice-based scheme that depends on the hardness of the *SIS* problem and is coupled with a private-key encryption scheme. Note that the encryption schemes used in both cases must be secure against *Chosen Ciphertext Attack (CCA)*.

[35] The IBM researcher mentioned in [84] (30/4/2025) that there is new Lattice-based research (accepted to appear soon) that produces ZK proofs and/or signatures with size O(log log n).