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

Shymaa Arafat (shar.academic@gmail.com, shymaa.arafat@gmail.com)

Abstract. The Estonian e-voting experience could be the richest to analyze; a country that is the pioneer in digitizing both the government and private sector since 2001, and hence digital voting in 2005, yet 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 Etonian IVXV i-voting system and propose some added security enhancements. The presented SoK includes applications implemented by election observers in 2023 & 2024 elections, which, to our knowledge, has never been mentioned and/or analyzed in the academic literature before. The paper also updates the general knowledge about an extra right given to auditors (but not observers) in the June 2024 European election, recent complaints, and about newer solutions suggested by academia in 2024. Finally, we discuss the current system status in 2024 EP elections and propose our own suggestions to some problems stated in the *OSCE-ODIHR* 2023 report that are still there.

Keywords: IVXV, El-Gamal Encryption, Verkle Trees, vote buying, counted-as-casted.

1 Introduction

Estonia is a small 1.37m citizen country located in east Europe who gained independence from the Soviet Union in 1991 and joined the European union in 2004 [1]. Most Estonian citizens welcomed the general *digital transition* in 2001; 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***[1] June **2023** report [2].

Although the ratio of citizens preferring i-voting is continuously increasing and has reached 51% in 2023, [3], one can trace a long history of rejection incidences from conservative right parties in [1,2 page8 footnotes16&17] from 2005 up till now; the situation was emphasized in 2023 when the internet votes flipped the election result for one of those parties, *EKRE*. The distribution of internet votes was completely different than that for poll station paper votes as shown in the curves in [4]; analysts due that to the society division mentioned above (ie., it's expected for the distribution to reflect the parties 'conventions on the voting method). Still, there were some complaining movements that continued persisting to the 2024 European elections [5].

[1] *OSCE* stands for Organization for Security and Cooperation in Europe, *ODHIR* stands for Office for Democratic Institutions and Human Rights; for IVXV to be used in voting for the European Parliament elections, European entities ought to approve its security.

Since most rejections come from right parties, a note about the effect of Ukrainian war on Estonia being the closest neighbor is appropriate; according to [6] there were Russian attacks on the system, but the authorities say it was properly defended[2]. On the political impact, we suffice with this article [7]; even if it was politically biased, it points out to the economic status with more refugees that the writer projects will seek Estonian citizenship, and thus can vote[3]. Also, some parts in the context of the *OSCE* report, [2, pages 15-16, footnotes 45&48], implies that Estonia is not very strict in giving citizenship for stable residents and their born children. The paper does not aim to dig deeper into political details; however, it is important to enlighten the reader about what seems to be the roots behind the split of opinions in the Estonian society.

Being aware of 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. Then section 4 explains in detail the current version of IVXV with important improvements that were done in 2023/2024, and section 5 discusses the remaining vulnerabilities of the system along with suggested solutions by the authors and elsewhere. Finally, section 6 gives the summary and conclusions of the presented work.

2 Recent opposing activities with technical merit

A technical incidence that gained some publicity in 2023 elections [8] was done by the same computer scientist observer in [4]; he voted using his own Python code [9,10], meaning that he *has overridden the official voting application* that voters should download to deliver their vote to the system. This gives an alarm that the voting application is not authenticated by the system, which was noticed by the *OSCE* report [2, page 8]; there were a mention of *some wrong district votes* too [11], but the report says they were corrected except one vote. It is the authors impression from all the listed references, that most submitted complaints were rejected based on passing deadline without objective investigation except only those from the mentioned observer where he was allowed to witness the processing of all his votes submitted through his code [8,9] in the 2023 elections; maybe because the incidence reflected an exploit in the system, and there were a doubt about his last valid vote too. A recent

[2] At the end of the presentation in [8], there was a question about the fears of Russian interference or taking advantage of the IVXV vulnerabilities.

[3] Although we used [1] as reference for the first sentence in section1, we wrote the updated number 1.374687m as of Jan 2024 from (https://news.err.ee/1609341741/statistics-estonia-s-2024-population-more-than-1-37-million) which mentions too that most of the annual increase (9000) are Ukrainians. Note also, that according to Wikipedia (https://en.m.wikipedia.org/wiki/Russians_in_the_Baltic_states) more than 63.7% (877,000 citizens) of the population are originally Russians, at least till Sep 2023.

complaint about *the decryption of invalid votes* after the current 2024 European elections (events are happening while we are writing) were also rejected objectively in [12]. 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 [13]. According to [14, Conclusion-page 60], generating proofs of correct decryption of invalid votes was remediated in code by the thesis writer to auditors only in 2024; however, *the file containing the decryption of invalid votes is only accessible to auditors* [14, page 22].

The recent European parliament elections has brought some newer actions from the i-voting opposing community [5,12,15,16,18]; the same observer mentioned above has developed some kind of shadow e-voting site they call *virtual threshold survey* [15] encouraging citizens[4] to vote again on it as a check. Also, a criticizing paragraph about IVXV mistakes from a researcher in *Cybernetica* TUT is being circulated online [16][5]; where *Cybernetica* (partnering with -*Smartmatic* [3,17]) is the company behind the current Estonian internet voting system since 2014 as we will detail in the following sections. However, in addition to the complaint in [12], we find the story of an earlier complaint about the election desktop also alarming; a first complaint granted the observer (on 23/2/2023) a 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 [19]. The observer took the photo shown in Fig.1 when the disk inspection took place (28/11/2023)[6]; it can be concluded from [18] that he pursued the matter further to the supreme court where they responded that "*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*"

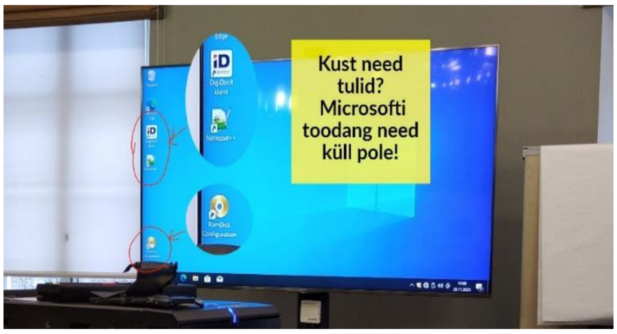


Fig.1: image taken from [18]; according to the observer, this is the computer used in key creation which was supposed to have an authentic Windows 10 operating system, but the

[4] We have no evidence of considerable participation ratio till the time of writing.

[5] The statement in [16] about "*a strategic mistake*" was made on March 2024, he made another (more moderate) statement on June (https://arvamus.postimees.ee/8036445/ago-samoson-selline-kontroll-parandaks-e-valimiste-usaldusvaarsust)

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

operator didn't even remember that DigiDoc4, Notepad++ and RamDisk tools were also installed on it.

Finally, we haven’t seen yet an *OSCE-ODIHR* report on the 2024 European elections, but another scientific report from ***the cyber security committee of the Academy of Sciences*** has been already handed to the election organizers [20]. Although the report is still under review and are not published yet, Google translating the minutes of the last committee meeting [21] on 3rd of June tell us they *have identified **6 threats** whose risk class is higher than small*.

3 A Brief on System Evolution

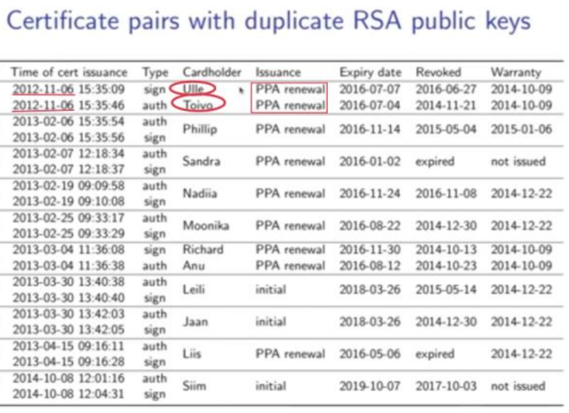
As mentioned earlier, digitization has been in Estonia for more than 20 years, since 2001, and has extended to include the private sector hand in hand with the e-government; e-ID cards existed since 2002, Fig.2, and electronic transactions is the casual behavior of the Estonian citizen. More details on digital system architecture and components like *Xroad*, *KSI* private blockchain is out of the scope of this paper and can be found in [22]; however, we find the e-ID key generation relevant since it is used in internet voting from its beginning in 2005 up till now. Hence, we will dedicate section 3.1 to one major event that changed a core cryptographic component of the e-ID system and i-voting as well, **RSA;** then we will follow with a brief on i-voting earlier evolution till it reached its main design as IVXV in 2017.

Estonian Electronic Identity Card (ID card)
EESTI VABARIIK
Republic of Estonia
ISIKUTUNNISTUS
Identity Card
JÕEORG
JAAK KRISTJAN
SPECIMEN
345678
IDESTAS12345671380010857l8<<<<
8001081M2001023EST<<<<<<<<<<<3
JOEORG<<JAAK<KRISTJAN<<<<<<<<<

Fig.2: Estonian eID card with 2 keys (authentication & signature), image taken from [1]

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. Such “rare” incidences where citizens were asked to re-install the Java Applet on the cards at PPA (the issuing authority) stations (otherwise the card transactions will be suspended after a certain time limit), became more frequent with time; hence providing more data & information for researchers to analyze, Fig.3 is a Table taken form [23].



Certificate pairs with duplicate RSA public keys

No	Time of cert issuance	Type	Cardholder	Issuance	Expiry date	Revoked	Warranty
1	2012-11-06 15:35:09	sign	Ulle	PPA renewal	2016-07-07	2016-06-27	2014-10-09
	2012-11-06 15:35:46	auth	Toivo	PPA renewal	2016-07-04	2014-11-21	2014-10-09
2	2013-02-06 15:35:54 2013-02-06 15:35:56	auth sign	Phillip	PPA renewal	2016-11-14	2015-05-04	2015-01-06
3	2013-02-07 12:18:34 2013-02-07 12:18:37	auth sign	Sandra	PPA renewal	2016-01-02	expired	not issued
4	2013-02-19 09:09:58 2013-02-19 09:10:08	auth sign	Nadiia	PPA renewal	2016-11-24	2016-11-08	2014-12-22
5	2013-02-25 09:33:17 2013-02-25 09:33:29	auth sign	Moonika	PPA renewal	2016-08-22	2014-12-30	2014-12-22
6	2013-03-04 11:36:08	sign	Richard	PPA renewal	2016-11-30	2014-10-13	2014-10-09
	2013-03-04 11:36:38	auth	Anu	PPA renewal	2016-08-12	2014-10-23	2014-10-09
7	2013-03-30 13:40:38 2013-03-30 13:40:40	auth sign	Leili	initial	2018-03-26	2015-05-14	2014-12-22
8	2013-03-30 13:42:03 2013-03-30 13:42:05	auth sign	Jaan	initial	2018-03-26	2014-12-30	2014-12-22
9	2013-04-15 09:16:11 2013-04-15 09:16:28	auth sign	Liis	PPA renewal	2016-05-06	expired	2014-12-22
10	2014-10-08 12:01:16 2014-10-08 12:04:31	auth sign	Siim	initial	2019-10-07	2017-10-03	not issued

Fig.3: First incidences of duplicate RSA keys whose owners were told to renew their ID cards at PPA stations; the table adopted from [23] with a detailed analysis of the marked with red case (they proved a valid signature for the first using the keys of the second)

Then, it was proven that the ID card manufacturing company, ***Gemalto***, generated the RSA keys outside the chip (could be to fasten the process) which violates the agreement rules and gives a chance for the key pairs to be copied and repeated. A lot of interesting details on how the analysis was done can be found in the presentation [23] and the paper itself [24]; more faulty keys issues[7] can be found in the author's PhD [25], and in another independent work [26].
According to [1], this was a global crisis for the company which was sued in many other countries around the world; Spain and Slovakia for example [27] replaced all the physical cards. Although Estonia did not replace the physical cards, they changed the company to ***IDEMIA*** [28] and the public key algorithm [29] moving to ***384-bit Elliptic Curve*** Cryptography ECC public key encryption[8].

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.
According to all available references, the main design theme of a <u>*double envelope protocol*</u> sending voter signed (first encrypted by the election authority key) ballot to the vote collector was there since <u>2005</u>. Then, based on [sec.1 of 30,31,32], we mark 2 milestone step transitions:

[7] Example errors include codes printed too dark which made them readable using torch, without opening envelope (happened twice 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] According to the official documents (page 14 in [33]) the "authorized voters list" is still signed using an 2048 bit RSA key.

- In 2011, a student named *Paavo Pihelgas*[9] demonstrated a proof-of concept ballot-manipulating software that relied on the absence of an acknowledgement from the vote collector to the voter that his/her vote was received. Hence, *the ability for voters to verify their votes* was first introduced in 2013. However, several flaws were discovered in 2014-2016; until *Cybernetica* partnered with *Smartmatic* to produce the QR verification code in its current form in IVXV.
- Then, with the appearance of other comparable e-voting systems (ex. *FLEP* in France & *SwissPost* in Switzerland), rich material was available for cryptographic research and lessons were learned. Hence, the 2017 and current version of the Estonian i-voting, IVXV, added a *vote-registration* service to guarantee no vote dropping, a *shuffling re-encryption mix-net* for vote privacy, and a non-interactive *zero-knowledge proofs of correct decryption* as will be detailed in the next section.

4 IVXV

In this section, we explain the design and structure of the Estonian internet voting system, IVXV, as described in the official documents [33]; we also detail two enhancements that happened in 2023/2024 before getting into the vulnerabilities of the current systems status in section 5.

4.1 Brief Factsheet

For a factsheet summary, the developing companies are ***Cybernetica-Smartmatic***; the voting device must be a desktop PC mobile voting is still postponed at least to 2025 [34]; *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.

[9] According to [32] the student filed a complaint to the Estonian Supreme court requesting to nullify internet votes in 2011 elections, but his complaint was dismissed (https://www.riigikohus.ee/en/constitutional-judgment-3-4-1-4-11); the verifying extension was only added in 2013 based on S. Heiberg & J. Willemson suggestion (https://ieeexplore.ieee.org/document/7001135)

4.2 System Architecture & Voting Steps

The system architecture and voting steps are depicted in Fig.4; the voting steps could be summarized as follows

1. The voter installs the voting application[10] , sometimes abbreviated as **VA**, on his/her PC.
2. After submitting eID (or mobile ID), the voting application checks the eligibility of the voter to vote through *the registration application*, **RA**, and if eligible displays the candidate choices for that voter (according to district if it's a local election).
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 *timestamp certificate*[11] received from the registration application, and then sends the double envelope ballot to the vote collector (sometimes abbreviated as **VC**).
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 (to be mixed and shuffled by mix-nets[12], then decrypted at the counting phase).
5. The vote collector sends a verifying ***QR code***[13] to the voter for optional vote verifying (through verification application) using a second smart device.

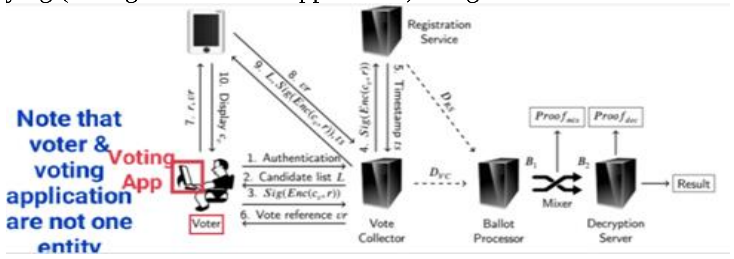


Fig.4: a diagram describing the architecture & the steps of the Estonian voting system, adopted from [1] originally in [33] (we added some colored remarks to remind the reader to distinguish between the voter machine and the voting application when it comes to attacks & vulnerabilities)

[10] Sometimes called the voter application in official documents, but we prefer to follow the naming convention in [31] to make it clearly distinguishable from the voter device.

[11] The timestamp certificate is important to distinguish the last vote of each voter, and also for checking the possible verify duration of 30 mins.

[12] IVXV uses *Douglas Wikström's Verificatum* (https://www.verificatum.org/); the package itself provides a verification application, and so does IVXV (and several other projects [30])

[13] According to [9], there were also a revealing incidence of *the president vote* through his QR code (when i-voted live in front of cameras, and showing his QR code, to encourage citizens to vote online; people took a snapshot of the QR code and revealed his vote). The incidence was mentioned in the context of doubting privacy and protection from coercion and/or vote buying.

4.3 Cryptographic Details of Last Fixed Attack

We find it significant, also gives a closer look into the used cryptographic primitives, to explain the exploit introduced in [31]; Fig.5 omits the voter signature and mix-nets parts and concentrate on the parts involved in the exploit and the introduced possible attacks.

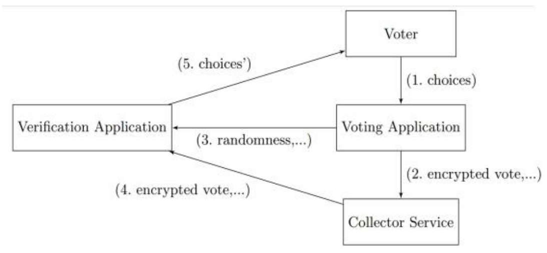


Fig.5: the part of vote casting and verification taken from [31]

Let the election public key be "**y**" with corresponding secret key "**S_k**", and "**g**" be the generator for El-Gamal encryption[14].

-To encrypt a vote "**v**" the voting application generates a random number "**r**", so that the encrypted vote is **$(C_1, C_2)= (g^r, \ y^r v)$**

-The verification application, working instantly within 30 mins, receives "**r**" from the voting application (hidden in the QR code) and calculates **$v = C_2/ y^r$** 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 (**S_k**) and the El-Gamal encryption known equation. **$y= g^{Sk}$** to calculate **$v= C_2/((C_1)^{Sk})$**

In the older design, the verification application only received C_2 from the vote collector; this gives a malicious voting application the chance to manipulate the encrypted cipher text by *sending wrong "r" value* to deceive the verification application (different values of C_1 for the same C_2).
Long story short, the authors found *three possible manipulations* with *a simple fix* of the vote collector to send the whole encrypted pair (C_1,C_2) to the verification application such that it also *verifies that **$C_1= g^r$*** as was finally done [35, lines 77-83 & 141-146 in code and the exception is thrown at line 60] on Feb 2023 just before March 2023 elections[15]. The authors alarmed that it is concerning [31, sec 3.6] that such a straightforward vulnerability wasn't noticed earlier, and then criticized the quality of IVXV in general [31, sec.4].

[14] We've mentioned in section 3.1 that a 384-bit elliptic Curve is used for the group field.

[15] One may find it suspicious that they didn't fix it when the authors of [31] first sent them a letter and waited till the last month before the election; clearly, the fix was after the time of writing because the authors stated: "*up till now the vulnerability was not fixed*" and that whenever they asked the reply was "*we're working on it*".

4.4 Range Proofs & Invalid Votes

Another improvement was added to IVXV on 30th May 2024 [36][16], just before the European Parliamentary elections on 3rd of June; invalid votes are 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 verify those proofs; that's why [14] suggested preventing invalid ballots from reaching the decryption phase at all. The thesis suggested the use of range proofs (based on *Bullet Proofs* & *Pederson Commitments*)[17] to check the validity of a vote, without revealing it, by the vote collector application; hence, the application should reject invalid votes and do not add them 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.

Why not reveal invalid votes?
As mentioned in section 2, there was a lot of debate and complaining about not allowing observers to view the decryption of invalid votes; however, the reasons stated by the state election service in the supreme court decision [12] do not match accurately those explained in [14].

- Reasons 1&2 in [12] talk about the technical infeasibility of decrypting invalid votes after the election and how this needs parts of the secret election key (issued only to members of the election commission); while [14] explains how the IVXV version used in 2024 already decrypts invalid votes in a separate file, and this can be traced in the opensource code [36].
- The 3rd reason in [12] of "*not knowing in advance what the invalid ballot contains and it may be an attack*" is rationalized better in [14] as the possible reveal of some information about the voter of the invalid vote, or more sever the threat of ***encoding attacks*** described in [37, sections 3.3 & 4.1] where an adversary can know the votes of several voters if able to submit an 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 [14] suggests.

5 Remaining Vulnerabilities/Issues

We do not aim to diminish the long Estonian experiment in i-voting that approaches 2 decades, but there are problems. Although there is not enough documentation yet to

[16] The GitHub in [36] shows that this version of IVXV is named 1.9.10-EP2024, while the previous fixed version in Feb 2023, [17], was named 1.8.2-RK2023.

[17] The authors of [14] preferred those as they are based on the Discrete Logarithm problem like El-Gamal encryption and considered general purpose SNARKs based on polynomial commitments not suitable for IVXV; a condensed summary of variations between possible SNARK choices can be found in the first 25 mins of (https://youtu.be/A3edAQDPnDY).

accurately trace all changes in the newer version, IVXV 1.9.10-EP2024[18], we know it is the one analyzed in the most recent report [21] which was done on collaboration with the election authorities, identified 6 threats with risk level higher than small; the *OSCE* 2023 report [2] pointed out to some issues too we have no evidence they're resolved.
As for academic research papers, in addition to pointing out attacks [31,32,37], many have introduced a holistic criticism to IVXV; [31] analyzed *IVXV public information* as *satisfying* only *1* (minimal restriction on disclosure of vulnerabilities) *out of 9 quality metrics*[19] and warned from the possible existence of hidden vulnerabilities; [37] demonstrated (through the analysis of possible privacy attacks) that *IVXV is vulnerable to attacks against vote privacy in those threat scenarios that were considered for it originally*; [38, sec. 5.1] discussed the different trust assumptions of IVXV including software components and key holders, we believe opensource parts of the system are not considered trusted and will elaborate more on the following subsections.

The IVXV team on the other hand sticks to the claim that there is no proved error in the election results; however, with *QR verification ratio of 5.5%*[20] as stated in the official i-voting statistics site [39], this does not really prove beyond reasonable doubt that no encryption pairs were manipulated.

In this section we discuss some obvious vulnerabilities in IVXV with possible solutions.

5.1 The Voting Application

The authors of [31] commented on the voting application being the only unrevealed part of IVXV code as an open source, while [38] identify this fact as a trust assumption. What is more of a threat is the incident of overriding it in 2023 election, [8,9], which proves that it is not even authenticated[21]; accordingly, the *OSCE* report [2] notified about the risk of not authenticating the voting application. A clear obvious vulnerability here 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 ~ 94% probability that the voter will not verify the vote, a ratio that could even increase by social engineering to target those who are not likely to verify, makes the risk level more severe.

[18] GitHub history shows 897 changed files with 34,059 additions & 10,830 deletions

[19] The quality standards are from an earlier, FC'21, paper by the same authors (*New standards for e-voting systems: Reflections on source code examinations*)

[20] The highest recorded verification ratio was 6.7% in 2021 elections, the statistics for the European 2024 elections are not available yet. It's not clear whether the ratio is per voter or per vote (in case of multiple voting); however, since total cancelled multiple votes is 10787 out of 313514 total i-votes (~ 3.4% with a max. of 8.6% in 2021 elections), we don't think this will make a significant difference.

[21] The one the observer developed was in Python, while IVXV code is written in Java.

A possible attack by a malicious voting application that could deceive even verifying voters was discovered by Olivier Pereira in [32]; *a malicious application could fake a system crash to* deceive the voter to vote again. This way the application will ***take the voter signature twice*** (generate two votes and two "*r*" values); hence while showing the voter the QR code of his/her choice, the system will consider it an old vote. Although the author suggested few mitigations, we have no clue that any of them was adopted. A simple solution, [40], for this specific problem is *to force a time interval between votes; the verification interval, 30 mins, seems a suitable choice*[22].

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[23] before transferring the money. This *DarkDAO*s idea was discovered by [41] in 2018 as a possible threat to decentralized voting in DAOs using governance tokens, but it could happen here too; the authors published a follow-up in 2023 [42] with a GitHub code.

֍ A general solution to all the above would be ***to authenticate the official voting application*** either through checking its *file digest* (hash SHA256 for its code for example) or assigning to it *a signature & authentication key pair* like the rest of involved applications in the system. However, another issue remains of ***how to inform a non-verifying voter that the vote was rejected because he/she has installed a malicious voting application***.
For example, [40], *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 (*PC interrupt*) to the voter screen. Although the authors of this paper are not very familiar with the technicalities involved in implementing this solution, we know hackers do it sometimes; if hackers can do it, then the vote collector application can a manage a way to do it.

5.2 Cryptographic Proofs

Another problem that was mentioned in the *OSCE* report [2] is that there's no cryptographic proof for *the deletion of multiple votes or ill-formed vote ballots*; i.e., the authorities are assumed trusted regarding not deleting or adding extra votes at this

[22] The authors of this paper sent a few suggestions to information systems emails from (https://www.valimised.ee/index.php/en/electoral-organizers/state-electoral-office/staff), and the time interval is the only one they considered "possible" in their reply. An appendix is added with the full emails (to be replaced with a reference, [40], to a PDF file when the authors names are revealed).

[23] Remote execution attests were originally discussed on Intel SGX (which is available on many new PCs in the market: https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions-processors.html), and it exists in other processor companies like Apple and others as well. So, we can assume there's a considerable probability that the voter PC can support it.

step. 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***". Again, this was viewed in [38] as trusting the vote collector and registration applications to not collude[24], otherwise it would be possible to drop ballots.
Meaning that, yes there are decryption proofs that what got into the *mix-nets* is exactly what got out of it and finally decrypted, and there is the possibility to design a public independent decryption proof verifier [30], but there is no cryptographic proof for the transition from the total list of votes to the "to be counted" list of votes.
The problem of *invalid votes* was discussed in section 4.4; thus, we concentrate here on the removal of multiple votes.

For this problem *we suggest to aggregate all votes in a 2-level Authenticated Data Structure ADS* [43, sec.2.1 def.3] which we can simply describe as data structures that can provide a succinct (short) cryptographic proof (sometimes called witness) of each element stored in them. In this paper, we suggest ***a Verkle Tree*** [44], which is a vector data structure that authenticate its elements based on KZG polynomial commitments for having the shortest verifying time, and because they provide a cryptographic proof of the number of elements stored in them (as opposed to Merkle Trees) which is beneficial in our case.
Hence, we suggest to aggregate all votes in a Verkle Tree, such that votes from the same voter are aggregated in a second level Merkle Tree[25], as in Fig.6.

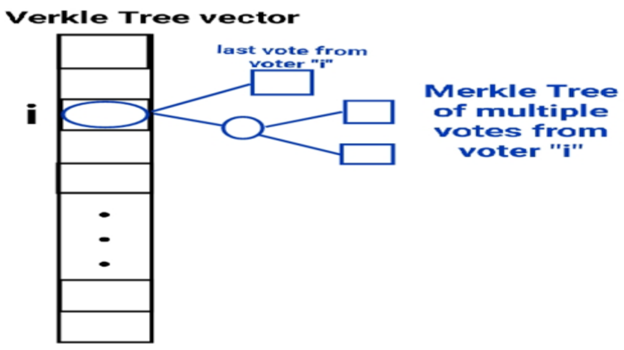


Fig.6: two-level Authenticated Data Structure, a vector that will be committed to using KZG commitments as a Verkle Tree, where each node value could be the root of a Merkle Tree containing multiple votes from the same voter

[24] Although Jan Willemson stated in his 2022 paper, [30], that the registration service makes it impossible to drop ballots, he gives a more accurate statement about the trust assumptions involved in his 2023 co-authored paper [38].

[25] We use Merkle (not Verkle) Tree in the second level, because we expect (in light of the available statistics) no voter will vote more than 8-16 times; ie, those subtrees rarely exists and are of 3-4 levels at maximum. It’s important to clarify that Verkle-Merkle is the suggested choice by the authors of this paper; other ZKPs still remain possible options, for example the STARK solution in [44].

To clarify more on the details:

-When a new vote enters the system (through the vote collector and the registration applications):

- ➢ If the attached voter signature hasn't appeared before, the new vote is added to the votes list, and is also aggregated to the Verkle Tree vector commitment.
- ➢ If it's a repeated signature, the previous vote is inserted to the Merkle Tree attached to the corresponding Verkle node and then the Verkle node is replaced by the new Merkle root and the new vote (the old node is deleted, and the new node is inserted).
- ➢ If the voter voted at the pulling station, its Merkle tree should still be kept for auditing/observation purposes; whether as a zero value with certain flag or in a separate Verkle Tree this could be an implementation decision.

-At the end:

- ➢ ***the number of nodes in the Verkle Tree proves the number of voters who voted (or valid counted votes)*** (*n* of a Verkle Tree is cryptographically proved).
- ➢ The counted votes are presented in the Verkle Tree nodes[26] (could be zero if voted at pull station) and their aggregation can be verified cryptographically.
- ➢ *Every deleted vote can be traced through its corresponding Merkle proof.*
- ➢ This way, the QR code can include the number of multiple votes for this voter, and possibly lead in the verification to a list of all previous multiple votes for that voter; this is an extra check against vote manipulation[27].
- ➢ Whatever the designer decision for handling ill-formed votes and those who voted at polling stations, the point is that ***old votes can be still traced*** and *if the code is open source all the numbers can be cryptographically verified*.

In general, this could be viewed as a way to prove that votes are *counted as casted* . We chose Verkle Trees since they provide a constant order complexity *SNARK* (per node) using KZG vector polynomial commitments, while traditional widely used Merkle Trees provides O(log n) proofs on the data stored in their leaves. On the other hand, STARKs introduced in [43] provides post-quantum security guarantee although

[26] A developer may choose to store the last vote twice (in the Verkle node and as the last leaf node in the Merkle tree); at this step of analysis, we believe it is an implementation decision.

[27] We know 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…, and your previous votes are". In fact, [9] shows, and demonstrated to the authors through X conversation(), that a sophisticated user can already do something similar through e-ID transaction confirmation service available in Estonia; however, or suggestion doesn't contribute to vote coercion since the voter can hide the latter QR code from the coercer.

having larger proofs; the authors also present projected performance analysis when applied to EL-Gamal based e-voting systems (but left the impact of mix-nets as a future work). Similar and comparative analysis of the Verkle Tree suggestion for the IVXV case is a possible future work.

6 Conclusions & Future Work

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 available resources to cover reported attacks and/or vulnerabilities and how they were fixed. Last but not least, we discussed remaining risks and unfixed vulnerabilities; mainly, authenticating the voting application and cryptographically proving the removal of "*not to be counted*" votes step. We suggested some possible solutions including checking the fingerprint or digital signature of the voting application and interrupting the voting process when it is wrong; also, enforcing a time delay between multiple votes to avoid fake crash attacks. Finally, we suggested the use of a 2-level authenticated data structure: *a* Verkle Tree to aggregate votes, where ***multiple votes from the same voter are to be accumulated in a Merkle Tree whose root is a node in the Verkle commitment***.

A possible future work is to conduct a comparative analysis between possible zero knowledge proofs that could be used; this includes comparing to the STARK approach suggested in [43] for Homomorphic e-voting schemes, and the somewhat contradicting opinion introduced in [14] that considered polynomial commitments-based SNARKs not a suitable fit for IVXV as it is based on the discrete logarithm problem.

Acknowledgements

I'm grateful to everyone participated in making me acquire this level of knowledge & research skills: from my graduation faculty Alexandria Computer Engineering department (the encouragement of *Prof. Nagwa Elmekky*, and the guiding soul of *Prof. Ahmed Belal*), to everyone who has put their work free online (Berkeley ZKP MOOC, HAL, iacr, arXiv, E-Vote-ID, Tallinn University). Also, a considerable part of the material presented here about observers came from *Märt Põdera's* blog, X (previously twitter) posts and conversations leading to more accounts & personnels.

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. 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://gafgaf.infoaed.ee/en/posts/great-divide-in-evoting/; last accessed 14/3/2024.
5. 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.
6. https://electionsnovascotia.ca/PictouWestByElectionEBallot, last accessed 22/4/2024.
7. Stone Bridge, "*About the destruction of Eestlus on the example of the Central Party*", https://uueduudised.ee/arvamus/kivisildnik-eestluse-havingust-keskerakonna-naitel/; last accessed 14/1/2024.
8. "*A computer scientist made available the code for e-elections, which the electoral service has so far been fiercely hidin*g", https://digi.geenius.ee/eksklusiiv/arvutiteadlane-tegi-kattesaadavaks-e-valimiste-koodi-mida-valimisteenistus-on-seni-kiivalt-varjanud/; last accessed 2/1/2024.
9. https://gafgaf.infoaed.ee/en/posts/perils-of-electronic-voting/; last accessed 4/1/2024.
10. https://media.ccc.de/v/37c3-12298-should_e-voting_experience_of_estonia_be_copied#t=965; last accessed 15/1/2024.
11. "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.
12. 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.
13. https://www.valimised.ee/en/internet-voting/observing-auditing-testing; last accessed 5/7/2024.
14. 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
15. "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.
16. 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.
17. Smartmatic-Cybernetica. IVXV Voting Service. Version 1.8.2-RK2023, https://github.com/valimised/ivxv/tree/master
18. "*E-voting system Disk appeared out of nowhere*", https://gafgaf.infoaed.ee/posts/esoteeriline-turvamudel/; last accessed 22/5/2024.
19. 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.
20. The report of *the cyber security committee of the Academy of Sciences*, https://x.com/danbogdanov/status/1802998209649762582; last accessed 6/7/2024.
21. Cyber Security Commission minutes of meetings, https://www.akadeemia.ee/akadeemia/noukogud-ja-komisjonid/kuberturvalisuse-komisjon/; last accessed 7/7/2024.
22. 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.

23. Arnis Parsovs, ”*Estonian Electronic Identity Card: Security Flaws in Key Management*"; video https://www.usenix.org/conference/usenixsecurity20/presentation/parsovs
24. Arnis Parsovs, "*Estonian Electronic Identity Card: Security Flaws in Key Management*", 29th USENIX Security Symposium, Aug 2020, 978-1-939133-17-5.
25. Arnis Parsovs, "Estonian Electronic Identity Card and its Security Challenges", PhD Thesis, University of Tartu.
26. 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/.
27. https://e-estonia.com/raulwalter-estonia-digital-identity-giant/; last accessed 20/3/2024
28. https://e-estonia.com/estonia-introduced-a-new-id-card/; last accessed 20/3/2024.
29. https://e-estonia.com/solutions/e-identity/id-card/; last accessed 20/3/2024.
30. 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
31. 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
32. Olivier Pereira, https://www.researchgate.net/publication/372570425_Individual_Verifiability_and_Revoting_in_the_Estonian_Internet_Voting_System
33. https://valimised.ee/sites/default/files/2023-02/IVXV-protocols.pdf
34. https://news.err.ee/1609194064/mobile-voting-likely-to-arrive-in-estonia-in-2025; last accessed 14/12/2023.
35. 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; last accessed 20/2/2024.
36. The key application, IVXV 1.9.10 EP2024, https://github.com/vvk-ehk/ivxv/tree/master/key; last accessed 8/7/2024.
37. Müller, J. (2023). “*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. Lecture Notes in Computer Science, vol 13412. Springer, Cham. https://doi.org/10.1007/978-3-031-32415-4_22
38. Krips, K., Snetkov, N., Vakarjuk, J., Willemson, J. (2024). *“Trust Assumptions in Voting Systems”*. In: Katsikas, S., et al. Computer Security. ESORICS 2023 International Workshops. ESORICS 2023. Lecture Notes in Computer Science, vol 14399. Springer, Cham, https://doi.org/10.1007/978-3-031-54129-2_18; full paper available at https://arxiv.org/pdf/2309.10391
39. https://www.valimised.ee/en/archive/statistics-about-internet-voting-estonia; last accessed 29/2/2024.
40. See the Appendix, to be added when authors names are revealed (after acceptance).
41. PMPhilip Daian, Tyler Kell, Ian Miers, and Ari Juels; July 2018; https://hackingdistributed.com/2018/07/02/on-chain-vote-buying/
42. James Austgen, Andrés Fábrega, Sarah Allen, Kushal Babel, Mahimna Kelkar, Ari Juels, "*DAO Decentralization: Voting-Bloc Entropy, Bribery, and Dark DAOs*", Nov 2023; https://arxiv.org/abs/2311.03530; https://github.com/DAO-Decentralization/dark-dao/tree/main; last accessed 20/3/2024.

43. 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
44. Zero Knowledge Berkely MOOC 2023, lecture 5, "*KZG polynomial commitment scheme*"; https://youtu.be/tAdLHQVWlUY

Appendix A

We present here an exact copy (just omitting the author's name) of the email thread with the IVXV team. We know this is too long to appear in the paper; if accepted, reference [40] will point to the PDF file in the authors GitHub containing both appendices A&B.

On Fri, Mar 8, 3:36 PM
to *sander.hyys, leino.mandre, arne.koitmae*

Dear Sirs,
As most researchers do, I'm obligated to send to you first before attempting to publish a research paper or article on your system.
Attached is the PDF file (Appendix B), and I'm including also the text of it in the email

---

On Thu, Mar 14, 1:26 PM
Indrek Leesi <indrek.leesi@valimised.ee>

Hello

Thank you for showing interest in our e-voting solution.

Enforcing a time limit between any two votes so that the voter can't be deceived by the very close timing

What should be the time limit I wonder, that can avoid the deception?

Make the Vote Collector/verification application check the OCSP used is the last one generated to this ID

I do not get the point of this one, because when the vote is arriving to server ocsp is being checked, same info is sent to verification app.

A possible safeguard could be to educate the voters to double check in case of an application crash through the myID service that they have signed only one vote, not two

Good suggestion. How many people will use it, is another question.

If there were 1)a file digest check (SHA256 of the code for example), or 2)an execution attest from the voter's device that the official voting applications was executed before accepting the vote, or 3)a signature key for all messages sent by the

Voting Application, like other applicationsin the system, we can be sure that votes from any edited version will be rejected, but is this enough???

All this - yes - not enough, we can make the voting app to send a shasum, use a key etc, but the voting application is downloadable executable app, anyone can decompile it, and make new app to use a key and send shasum for collector to accept the vote. But we have the app chasum on our website to verify the app, that people actually use often, and because it is digitally signed it cant be faked.

There must be a way to inform the voters if their vote was rejected due to bad voting application, even if they did not try to verify it. Besides, I understand that the Estonian e-voting system may deliberately choose to allow anonymous voting applications to increase public trustwith the code of the official one being a secret5. Hence, I can think only of solutions depending on either 1)allowing the vote collector application to send messages directly to the voter screen (not through the voting application), 2)or by using a second device (a text msg to the voter's mobile *for example). Then, we can do one of the following approaches:*

1) I can't understand how could collector send message to some computer screen - in what way, how?
2) Second device - as mobile - how could we get the phone number of a voter? Mobile numbers are not available to Election office. Also quite many use prepaid cards. Even if we make a mandatory to register as a voter and add a phone number - people can change their phone numbers or make mistakes by inserting - anyway it is quite a mess.

1-Authenticate the voting application in the way you prefer; if the vote is not coming from it, reject the vote and inform the voter.

Same case as few blocks above - decompiled and remade voting app can mimic the authentication

2-To keep things as they are (official VA not revealed, voting using anonymous Voting Application is possible), I suggest calculating and checking the file digest or hash of the used application and then treating anonymous voting applications as Ballot Marking Devices

Same as point 1, self made app can mimic the digest

3-A third way could be to require vote applications to register earlier same as point 2

Finally, a third remark is about providing global public checks. The verification application only provides individual verifiability, and for interested voters only, however it doesn't provide universal verifiability.

What is universal verifiability - in technical - how can this be achieved. Currently auditor verify every step - on that we relay on. Auditor also check that all chasum of the votes will end up to mixing, so to be sure no cryptograms are not being changed - "ecorded as casted".

1-Don't wait for the voter, invoke the verification application always by the vote collector application; again to either display a confirmation directly on the user screen or text it to the user mobile. Then wait for a back confirmation from the voter; I think this way individual verifiability is 100% achieved

If we need to find malicious voting app-s then this is won't help, tampered app will show whatever and confirmation will also being sent automatically. And what comes to phones, again - we do not have voters mobile numbers.

2-A simple added step to the registration and/or vote collector application is to accumulate a Zero Knowledge Proof on all votes to be compared to the whole list of all votes

If people do not trust digitally singed votes then the 0-proof does not convince them either.

3-I believe there could be a simple way for the registration and/or vote collector applications to accumulate all votes from one voter in a single term of a second KZG polynomial...........

What is the idea or goal behind this?

Indrek Leesi
Riigi valimisteenistus, Toompea 1, Tallinn
indrek.leesi@valimised.ee

On Sat, Mar 16, 2:22 AM

Authors replying back

First, thank you Sir for replying. Then about the replying comments:
1-About the **time interval between votes**; your same choice for a possible verification (30 mins in your case) seems the most suitable choice so that verifications do not overlap. Note that a rejection of any edited version of the voting application will prevent the problem; an automatic verification (even if the voter didn't request it) will also solve this specific problem since the verification application will automatically send the verification of the last vote.

2- About authenticating the voting application, there is a mandatory disagreement point here; ***if you keep a file digest or a signature key for the voting application, any edited & recompiled voting application will definitely 1)have a different hash value, 2)will not know the secret key you assigned to the official one, and 3)will not be able to deliver the execution attest from the voter desktop.***

i.e., **yes indeed you can reject votes from other voting applications** than the official one; what I meant when I said this is not enough to solve the problem is that we must inform the voter that the vote was rejected due to failed authentication of the voting application, otherwise the voter may never know and lose his/her vote.

3-About how, I mean the verification application should find a way to get the IP address of the voter desktop and sent a direct message to the voter screen; ie, definitely not through the doubted voting application. This level of technicality is not my area, but I know if hackers can send direct messages to a person's device then a specialized developer can find a way to do it when receiving data from an application running on that machine.

4-The point from providing a Zero Knowledge Proof of the aggregated votes is:

a) To solve the last problem mentioned in the OSCE report; ie, making the counting process trustless and provide confidence that no votes were added/deleted/edited during the step of removing repetitions and bad votes. Quoting from the OSCE report "*The process of verification of shuffling of ballots to ensure that the encrypted votes are fully separated from the identity of voters and the verification of the decryption of ballots was conducted successfully on 6 March, the day after the elections. However, the ODIHR EET observed that* ***the critical step of removing the votes overwritten by another vote cast on the internet or in a polling station was not audited"***
and the accompanying footnote saying: "***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".***

b) Most complains and rejecting voices I've read against i-voting stress on observability and auditability, those public proofs with their code open sourced will provide auditable proof for every step.

c) The idea of doubting ZKPs differs between countries; I understand Estonia is digitized and Estonian citizens are used to cryptographic proofs in all aspects of life, not just e-voting since 2005; plus that I understand i-voting and IVXV will remain in Estonia anyways, we are just trying to fix its vulnerabilities. Frankly, this was a strange sentence that seems out of context to be said by one of IVXV team???

5-The example ZKP I suggested is keeping a Verkle Tree of all votes, such that multiple votes from a certain voter are represented by a Merkle tree of all the voter's multiple votes (assuming no one will vote more than 8-16 votes the Merkle Tree attached to each term of the Verkle Tree will not grow so much). By making this code open source anyone can run the ZKP

On Thu, Mar 21, 3:51 PM (A second enquiry by the paper authors)

Dear Sirs,
In addition to expecting a 2nd reply from you after I clarified my suggestions more accurately and maybe explained in detail; I have one simple question:

I have read that after the RSA manufacturing problem Estonian has moved to Elliptic Curve Cryptography with a company named IDEMIA since 2018. However, I've read in an i-voting official pdf document (dated 2022) that the eligible voters list is signed using a 2048 bit RSA key; so, is that still the case or it's just a couple of lines that missed the info update?

Thanks & Regards

Appendix B

The file sent in the first email:

Dear Sir,

In the process of preparing a SoK paper (Systemization of Knowledge) on the Estonian internet voting system, naturally I've read [1,2,3] in addition to the concerned scientific papers like [4,5]. I have learned the following that I think I'm obligated to send to you, the team responsible on IVXV, first before attempting to publish any research paper about your system:

-First, a short note on the revoting after application crash ***attack discussed by Olivier Pereira*** in [5][28]; I'm aware that the paper author has contacted you and that his paper contains many mitigation suggestions. However, let me suggest two more:

[28] And mentioned also in the OSCE report (https://osce.org/files/f/documents/f/f/551179_0.pdf)

1-Enforcing *a time limit between any two votes* so that the voter can't be deceived by the very close timing,

2-Make the Vote Collector/verification application *check the OCSP used is the last one* generated to this ID.

3- A possible safeguard could be to educate the voters to double check in case of an application crash through the *myID service that they have signed only one vote*, not two[29], in this time interval. However, I wouldn't recommend relying on voters to double check integrity when only 5.5% of them invoke the verification application [6].

. . .

-Secondly, the previous problem is only a subset of the main problems surrounding ***the Voting Application***, which is the core of this note:

In addition to most papers criticizing the IVXV voting system for not revealing the Voter Application code to the public as an open source; a real risk became evidence clear through the incidence in [1] which proves that *IVXV does not authenticate the voting application cryptographically* before accepting votes from it. This fact was mentioned in the OSCE ODIHR report too; the Olivier Pereira attack wouldn't be possible without such vulnerability either[30]. The "skilled voter" attempting his own code is not the only risk, a major risk is an innocent voter deceived to download a malicious version instead of the original official voting application; another risk is large scale vote buying applications that could be fixed to certain vote choice like Dark DAOs introduced in [7,8,9] where money is paid only when receiving an execution attest of the vote buying application from the voter machine[31]. In short, this leaves the door open for adversaries to try their best to fake or change votes.

Suggested Solutions:

-If there were 1)*a file digest* check (SHA256 of the code for example), or 2)*an execution attest* from the voter's device that the

[29] Yes, the QR code contains a timestamp, but in this case where the difference in time is pretty small; a voter may just say to him/herself "Oh then, the system calculates the time from the step before the crash". However, thru myID the voter will realize that there are two signed votes.

[30] Since it can only be done by a malicious voting application

[31] I understand that IVXIV allow only voting from Desktops, where I assume the functionality of execution attest is possible.

official voting applications was executed before accepting the vote, or 3)*a signature key* for all messages sent by the Voting Application, like other applications in the system, we can be sure that votes from any edited version will be rejected, but is this enough???

-There must be a way to *inform the voters if their vote was rejected due to bad voting application,* even if they did not try to verify it. Besides, I understand that the Estonian e-voting system may deliberately choose to allow anonymous voting applications to increase public trust with the code of the official one being a secret[32]. Hence, I can think only of solutions depending on either 1)*allowing the vote collector application to send messages directly to the voter screen* (not through the voting application), 2)or by *using a second device* (a text msg to the voter's mobile for example). Then, we can do one of the following approaches:

1-Authenticate the voting application in the way you prefer; if the vote is not coming from it, reject the vote and *inform the voter*.

2-To keep things as they are (official VA not revealed, voting using anonymous Voting Application is possible), I suggest ***calculating and checking the file digest or hash of the used application and then treating anonymous voting applications as Ballot Marking Devices*** (applying similar approaches to those used for detecting malicious or malfunctioning BMDs), *but after warning the voter*[33]. For example, after warning the voter, introduce a Benaloh challenge by asking the voter to vote for a random option chosen by the Vote Collector first before accepting the actual vote. Another auditing recommendation is to keep the file digest of all used voting applications to perform final checks if all votes from that application goes to a fixed candidate[34].

[32] I mean giving the message "if you don't trust ours, feel free to develop your own", since I have noticed that the OSCE report when notifying such risk, Sec V page 6, did not stress on the necessity of using the official voting application as the only solution.

[33] Assuming three possibilities for the voter in this case: 1-a voter deceived with a malicious application will quit and re-install or vote in polling stations, 2-an opposition voter not trusting the official one and wants to vote with another code will continue the voting process and accept the challenge, 3-a coerced or vote selling voter will tell them I can't use your application because fixed choice applications are rejected.

[34] This is for auditability or data analysis reason. For example, if an application is developed by a certain party and maybe published on its site, it would normal if all votes coming from it goes to that party; however, there's no way to distinguish this from coercion or vote buying if such party has money and power.

3-A third way could be to require vote applications to register earlier enough before election to investigate their code and if approved store their file digest or an authentication key along with that of the official application. Then at election time allow only registered applications[35].

. . .

-Finally, a third remark is about providing global public checks. The verification application only provides individual verifiability, and for interested voters only, however it doesn't provide universal verifiability. When the official site of Estonian i-voting statistics [6] shows that the actual QR code verification ratio was 5.5% in 2023 elections, and reached a maximum of only 6.7% in 2021; this raises reasonable doubt and I think is not enough for end-to-end verifiability. The doubts in the OSCE report concerning the multiple & bad votes removal step is another issue; I also believe that the decryption proof is not enough for universal verifiability. It proves what entered the mix-nets is what came out of it but does not prove the integrity of the input; I may call that proving "*counted as recorded*", but *not "recorded as casted"*, assuming the verification application proves *"casted as intended"*.

I can think of a few mitigation steps:

1-*Don't wait for the voter, invoke the verification application always* by the vote collector application; again to either display a confirmation directly on the user screen or text it to the user mobile. Then wait for a back confirmation from the voter; I think this way individual verifiability is 100% achieved.

2-A simple added step to the registration and/or vote collector application is to ***accumulate a Zero Knowledge Proof on all votes*** to be compared to the whole list of all votes *before and after filtration*; I thought of a *verkle tree*[36] (Vector Commitments using KGZ polynomial commitments) of all votes, and there is a recent paper [11] that suggest using *STARKs* to prove tally votes are "counted as casted". *The ZKP values* could be *published* on the KSI blockchain available in the

[35] Although this may comfort doubting political parties, to design their own application, I think this approach may cause public mess or raise more doubt; it also may exhaust the system managers with reviewing so many applications. I only mentioned it because it is feasible and achieves the purpose.

[36] In addition to the shorter proof size, Verkle Trees has the advantage (over Merkle Trees I mean) of cryptographically proving *n* the number of values which is of great importance as it represents the number of votes or voters in our case.

digitalized information system in Estonia, or written in any public Bulletin Board, or strong credibility blockchain like Bitcoin/Ethereum/Algorand/…; as long as *the publishing code is part of an open source application*. With such values publicly available, and the code calculating them is also publicly available as an open source, I think this will provide universal verifiability and increase public and political trust in the system.

3-I believe there could be a simple way for the registration and/or vote collector applications to *accumulate all votes from one voter in a single term of a second KZG polynomial* (through delete and re-insert) for the cryptographically proved *n* to *represent the number of voters*. However, I will wrap up this letter without getting into such details because I have read that there are European elections coming soon, and the sooner you receive my humble note is the better.

4-I can't think of a way to cryptographically prove that bad votes are bad votes.

References

[1]https://media.ccc.de/v/37c3-12298-should_e-voting_experience_of_estonia_be_copied#t=965, last accessed 15/1/2024.

[2] https://gafgaf.infoaed.ee/en/posts/perils-of-electronic-voting/, last accessed 15/1/2024.

[3] https://ausadvalimised.ee/docs/yhisavaldus2023/ ; and their GitHub link, https://github.com/vaatlejad/vaatlejad.github.io

[4]https://www.researchgate.net/publication/376231158_On_the_Auditability_of_the_Estonian_IVXV_System_And_an_Attack_on_Individual_Verifiability

[5] Olivier Pereira, "Individual Verifiability and Revoting in the Estonian Internet Voting System", Aug 2021; https://eprint.iacr.org/2021/1098

[6] https://www.valimised.ee/en/archive/statistics-about-internet-voting-estonia; last accessed 29/2/2024.

[7] PMPhilip Daian, Tyler Kell, Ian Miers, and Ari Juels; July 2018; https://hackingdistributed.com/2018/07/02/on-chain-vote-buying/

[8] James Austgen, Andrés Fábrega, Sarah Allen, Kushal Babel, Mahimna Kelkar, Ari Juels, "*DAO Decentralization: Voting-Bloc*

Entropy, Bribery, and Dark DAOs", Nov 2023; https://arxiv.org/abs/2311.03530 ; https://github.com/DAO-Decentralization/dark-dao/tree/main

[9] Mahimna Kelkar, Kushal Babel, Philip Daian, James Austgen, Vitalik Buterin, Ari Juels,
"*Complete Knowledge: Preventing Encumbrance of Cryptographic Secrets*"[37], May 2023; https://eprint.iacr.org/2023/044

[10] 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

[37] Thought it maybe relevant since it suggests a possible solution, at least for Android, and I know there are future plans for mobile voting