

Some Notes on The Estonian IVXV internet Voting System

Shymaa M. Arafat

Shar.academic@gmail.com, Shymaa.arafat@gmail.com

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]¹; I'm aware that the paper author has contacted you and that his paper contains many mitigation suggestions. However, let me suggest two more:

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 two², 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 either³. 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

¹ And mentioned also in the OSCE report (https://osce.org/files/f/documents/f/f/551179_0.pdf)

² 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.

³ Since it can only be done by a malicious voting application

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 machine⁴. 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 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 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-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 voter⁶. 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 candidate⁷.

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 applications⁸.

⁴ I understand that IVXIV allow only voting from Desktops, where I assume the functionality of execution attest is possible.

⁵ 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.

⁶ 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.

⁷ 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.

⁸ 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.

. . .

-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 tree⁹ (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 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.

Acknowledgement

This note was prepared and evolved through the discussions with (Märt Põder / @tramm@mstdn.social) as shown in these twitter (X) threads

(<https://twitter.com/trtram/status/1765300578131808257>). I needed to contact a real Estonian

⁹ 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.

voter to get the complete picture, and he is also an observer with the special experience of voting using his own coded voting application; some of the things that should be systemized and added to the literature survey of the Estonian e-voting experiment.

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"¹⁰, 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

¹⁰ Thought it maybe relevant since it suggests a possible solution, at least for Android, and I know there are future plans for mobile voting