Removing Insiders' Trust from The Estonian Internet Voting System (IVXV)

EVOTeID 2025 International Conference

Tenth International Joint Conference on Electronic Voting

1-3 October 2025 - Nancy, France

Shymaa M. Arafat
Associate Professor (Independent, https://github.com/DrShymaa2022)
shar.academic@gmail.com, shymaa.arafat@gmail.com

OSCE/ODIHR in Feb 2025: "*Political parties historically opposing e-voting currently have 4 main areas of concern*", the first 2 are

E2EV & Protection against internal threats (insider attacks) which was explained in their 2023 report as

"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"

=> We propose 2 alternative solutions

Consistency Checks:

Since both use the same timing service, PKIX, generalize what voters using eID can currently do as a double check for individual verifiability using the existing *myID* service:

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

-The same could be repeated for checking the integrity 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)
```

-The first check could be instead accompanied by some kind of *Risk Limiting Audits (RLA)*s, where only a sample of random votes could be selected to check manually. Here, TXs stored in the Estonian information system will 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);
}
```

Cite as:
Shymaa M. Arafat, "*Removing Insiders' Trust from The Estonian Internet Voting System (IVXV)*", E-Vote-ID 2025, Track-4 Posters and Demos, 10th International Conference on Electronic Voting, 1-3 Oct 2025, Nancy France.

Although [3] has introduced integrity checks to remove trust in the Ballot Process, technically IVXV trust assumption admitted in [4] remains; *the Vote Collector (VC) and the Registration Service (RS) are not to collude together*. A recent paper [5/table1] that applied automated formal verification tools on IVXV has reached a similar result; the system fails to provide its goals (integrity and privacy) if 2 out of 3 colluded (VC, RS, and the timestamping service TMS). In this poster, we propose two alternative solutions for the Estonian System to eliminate insider attacks.

-The first is to use different queries and/or statistical techniques (like RLAs [6]) to *check the consistency of multiple sources of information that already exist in the Estonian government*; digital IDs activity logs where myID service [7] is an example.

-The second is to *aggregate votes online in an Authenticated Data Structure that cryptographically proves the number of values* stored in it (the number of votes in our case); we suggest the use of Verkle Trees [8] for their fast proof generation time as benchmarked (independently by someone else) in [9] to have time ~ 1 second for n < 2^{22}~ 4 million. (on Windows Intel i5-4690K, 22GB)

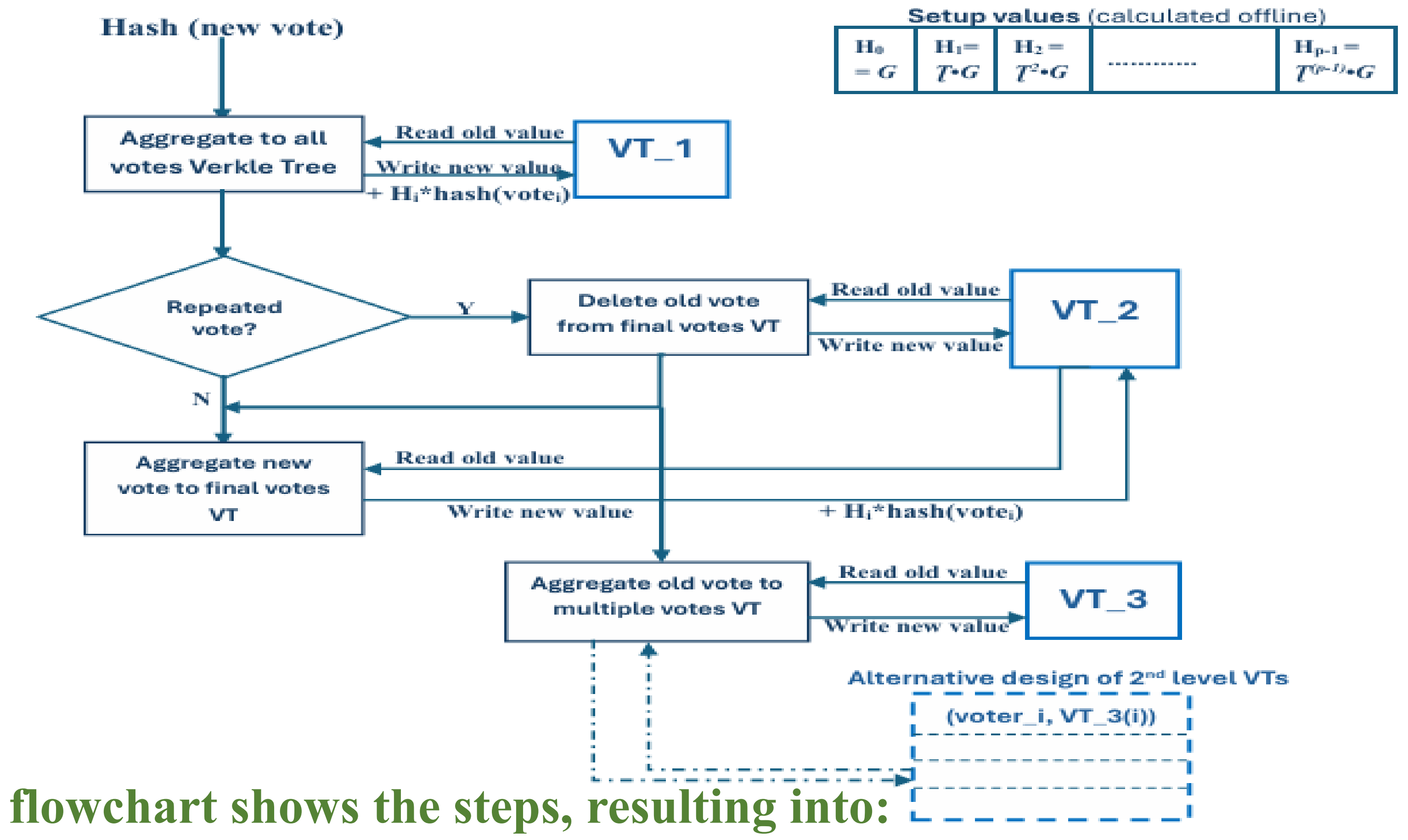
1. OSCE/ODIHR, "*Estonia Parliamentary Elections*", 5 Mar 2023, https://osce.org/files/f/documents/f/f/551179_0.pdf (p.10/ft.22)
2. OSCE/ODIHR, "*Opinion on the Regulations of Internet Voting in Estonia*", 17 June 2025, https://osce.org/files/f/documents/e/a/593435.pdf (quoted part from sec.3/28)
3. T. Treier and K. 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
4. 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
5. S. Baloglu, S. Bursuc, S. Mauw, and J. 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
6. Risk Limiting Audits (RLAs), "*What is RLAs*", https://verifiedvoting.org/audits/whatisrla/; "*RLAs Frequently Asked Questions*" https//www.sos.state.co.us/pubs/elections/RLA/faqs.html
7. https://myid.skidsolutions.eu/en
8. Zero Knowledge Berkely MOOC 2023, lecture 5, "*KZG polynomial commitment scheme"*; https://youtu.be/tAdLHQVWlUY
9. 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
10. **https://github.com/DrShymaa2022/SoK_Estonia_IVXV_EVoteID/blob/main/Estonia_EVoteID_long.pdf**
11. M. Harrison and T. 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
12. 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

Verkle Trees:

-Verkle Tree, [8], 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 (the number of votes in our case). 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++;*} , the vector H is calculated in the setup phase and $\mathcal{T}$ secrecy is critical

$$H_0= G,\ H_1=\mathcal{T}\bullet G,\ H_2=\mathcal{T}^2\bullet G,\ \dots,\ H_{p-1}=\mathcal{T}^{(p-1)}\bullet G$$

(Time comparisons with Merkle + Plonk in [9], discussion on [10] for Verkle Trees vs. STARKs [11] or Merkle Trees. Finally, [12] suggested a ZKP to every vote in 2022 to defend vote privacy attacks by insiders.)



-The flowchart shows the steps, resulting into:

- ➢ **Combined with [3,9/5.2], we may use *only one VT* to prove the number of recorded votes**
- ▪ **Verify: *n(VT_1)= count(votes list)***

OR:

- ▪ **Verify: *n(VT_1)= n(VT_2) + n(VT_3)***
- ▪ **Verify: *n(VT_end) = n(VT_2) – n(VT_4 + VT_5)***
- ➢ **VT_3 [i] can prove the number and values of every deleted (multiple) vote for the sampled voters**
- ➢ **In the 2-levels VT_3 case, QR codes can include the number of multiple votes for each voter.**