Automated Ballot Stuffing with an Encrypted Vote: A Large-Scale Attack on the Estonian Internet Voting System (IVXV) and its Mitigation

EVOTE ID 2025 INTERNATIONAL CONFERENCE

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

Tenth International Joint Conference on Electronic Voting
1-3 October 2025 - Nancy, France

Malicious Voting Application + Online Auctions + Execution Attestations => Cloning a vote you don't even know (to be determined at runtime)

We put alarms on how online coding via smart contracts can add new threats to e-voting, and how severe could be not authenticating the Voting Application for IVXV as OSCE/ODIHR warned (it is not just the Pereira attack anymore)

Steps:

1-The adversary writes a malicious Voting Application VA' then uploads it to a designated site for targeted voters to upload.

2- Coerced/bought voters, and devices that possess compromised voters credentials bought from dark web [1] download VA'.

3-The adversary decides (or receives from an anonymous) the desired vote "*v*", encrypts it as $(g^r, y^r.v)$, then writes it in a predefined memory location.

A more sophisticated option for 3:

3.a - The adversary writes a smart contract to perform an *online auction between bidders* to select the winning vote "*v*", encrypt it as $(g^r, y^r.v)$, then write it in the predefined memory location.

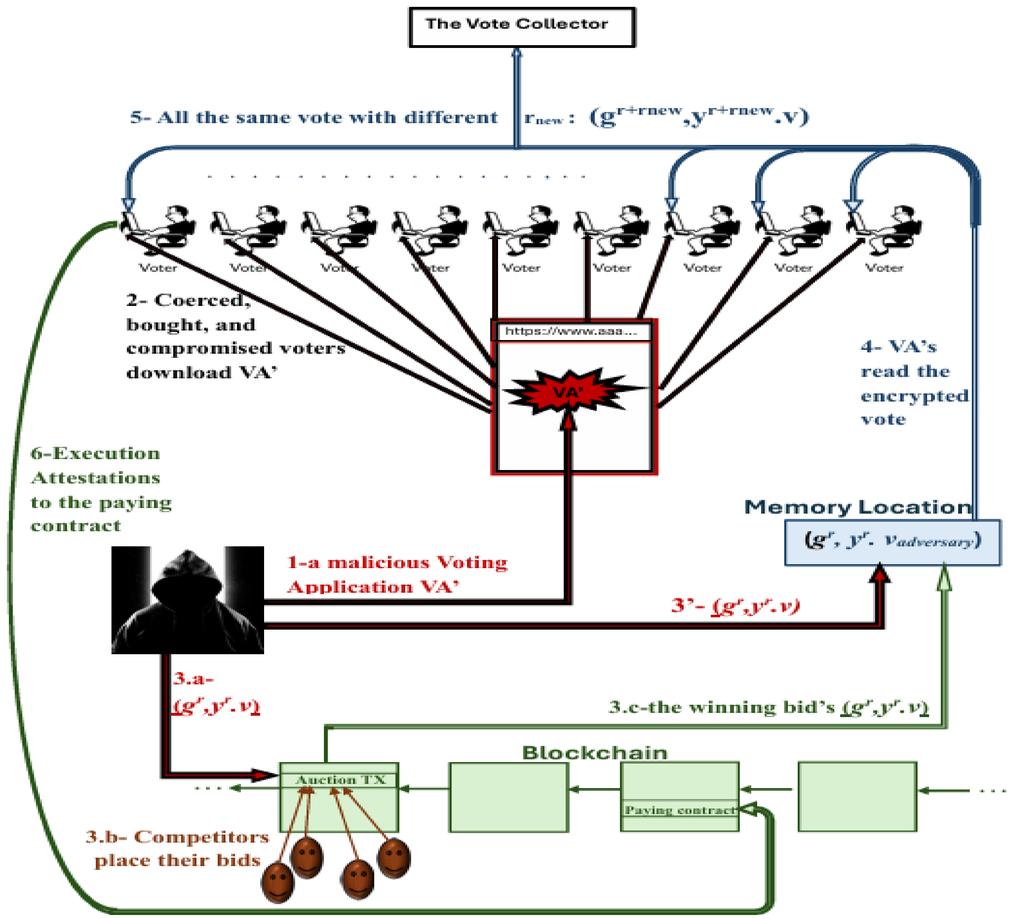
3.b -Competitors (possibly candidates or maybe other countries) place their bids for the online auction. Another variant may also allow voters to auction their votes for the highest buyer.

3.c The smart contract writes the winner's desired value $(g^r, y^r.v)$ into the predefined memory location.

4-The malicious VAs running at voter devices read the encrypted vote from the memory location at a designated time.

5- Each VA generates a fresh random number say r_{new} to hide the encrypted vote as $(g^{r+rnew}, y^{r+rnew}.v)$, then sends it to the Vote collector as the voter's choice.

6- Each voter's device sends an *execution attestation* [2] of running VA' *to the paying contract* (another smart contract) to prove that the order has been obeyed and possibly get paid.



1. *Dark web examples*: 1. https://www.bitsight.com/blog/what-are-compromised-credentials, https://www.theguardian.com/world/2023/feb/15/revealed-disinformation-team-jorge-claim-meddling-elections-tal-hanan, https://www.olfeo.com/en/les-serveurs-command-control/, https://www.computest.nl/en/knowledge-platform/blog/arrests-worldwide-genesis-market-for-online-identity-fraud/, https://www.experian.com/blogs/ask-experian/heres-how-much-your-personal-information-is-selling-for-on-the-dark-web/, https://www.politie.nl/en/information/checkyourhack.html
2. *Dark DAOs*: https://hackingdistributed.com/2018/07/02/on-chain-vote-buying/, https://www.youtube.com/watch?v=DFdD8qibFi4, https://github.com/DAO-Decentralization/dark-dao/tree/main
3. https://bitcoinelectrum.com/how-to-verify-your-electrum-download/
4. https://github.com/DrShymaa2022/SoK_Estonia_IVXV_EVoteID/blob/main/Grok_X_Electrum_1button_IVXV.pdf , https://github.com/DrShymaa2022/SoK_Estonia_IVXV_EVoteID/blob/main/Estonia_EVoteID_long.pdf
5. Olivier Pereira, "*Individual Verifiability and Revoting in the Estonian Internet Voting System*", https://eprint.iacr.org/2021/1098
6. Jan Willemson, "*Recommendations to OSCE/ODIHR (on how to give better recommendations for Internet voting)*", 10 Feb 2025, https://arxiv.org/html/2502.06385v2
7. *OSCE/ODIHR* 2023: https://osce.org/files/f/documents/f/f/551179_0.pdf, 2025: https://osce.org/files/f/documents/e/a/593435.pdf
8. https://a16zcrypto.com/posts/podcast/quantum-computing-what-when-where-how-facts-vs-fiction/

Mitigations:

-The solution *is to authenticate the official Voting Application*; the following suggestions all fulfil the current system philosophy of giving suspicious voters the freedom to use another voting application they trust more.

1-Publish the Voting Application *file digest* (ex.: hash SHA256 for its code) and encourage voters to run a check when downloading it *like the Electrum Bitcoin wallet* [3] but batched into 1 button click [4] to make it easier for voters. Has more usability and solves the Pereira attack [5] more robustly than depending on the OS verifying a developer key signature as [6/sec. 4]; still voter dependent and doesn't help dark web compromised voter credentials.

2-Assign a signature & authentication key pair to the official voting application like the rest of applications. Then the election authority can *allow only usage of pre-registered private voting applications with a stored public key at the voting server*; the election authority can also *scan them for any malicious code* before granting usage rights. Two issues: providing enough guarantees that authorities will be flexible in allowing opposition VAs given previous problems [7]; informing voters if their vote was rejected for using a non-registered VA.

-Both solutions could be post-quantum by *using post-quantum hash based digital signatures*, like Stateless Hash (SLH)-DSA approved NIST standard, since its relatively large size of 7k Byte is only transmitted once with the download and won't be noticeable by users [8/min 64].

Extra Safeguards:

-Add a *humanity check* for the Vote Collector to be sure the interaction process with VA is not automated (Swiss system).

-For QR-code checkers, the verification application could display an extra message "*You voted using (the official/a different) voting application".*

-Add a *"Reject all"* option to make boycotters vote to minimize the effect of stolen credentials.