Design and Build A Secure E-voting Infrastructure

Ahmed Hassan
Department of Computer Science
College of Staten Island, CUNY
2800 Victory Blvd, Staten Island, NY 10314
Ahmed(at)Linuxism.com

Xiaowen Zhang
Department of Computer Science
College of Staten Island, CUNY
2800 Victory Blvd, Staten Island, NY 10314
Xiaowen.Zhang(at)csi.cuny.edu

Abstract — **Our objective is to build a fully functional remote e-voting system that can be used in local, state and federal elections. We have designed a secure online e-voting system that provides confidentiality, anonymity, integrity, authenticity, auditability and verifiability. Confidentiality prevents anyone else from knowing who has voted for whom except the voters themselves. Anonymity is to prevent the tracking of the voters' real identities. Authenticity is to ensure that voters are eligible with unidentifiable-untraceable signatures of the real votes. Integrity is to ensure that no one else is able to change the ballot, and to detect the change if it occurs. Auditability makes the election auditable with or without tracing back the true identity of the voters. Finally, verifiability enables the system to verify and count the votes, and to detect any missing or fraudulent votes.**

Keywords: E-voting, Internet voting, election,cryptography.

I. INTRODUCTION

Electronic voting and Internet voting are an active research area. Entering "electronic voting" in Google search, 10 million search queries will be returned. A huge number of top tier computer scientists study the area of voting every year. For instance, since 2006, USENIX hosts EVT/WOTE annual conference (Electronic Voting Technology Workshop/Workshop on Trustworthy Elections). Currently, there are over 300 papers has been published in that conference alone. However, the Internet based e-voting is being considered a hard problem among academic researchers and industrial practitioners. The reason is, partially, because of the not-so-good previously implementations of e-voting systems. Considering the current status of the Internet almost all mainstream researchers are generally not in favor of the Internet based e-voting. They believe there will always be a weak point that can be exploited to attack an election. We will review some of these systems, point out the weakness and security properties, and design a good system from the scratch.

Definition of a Secure E-voting System: A new practical secure e-voting scheme should satisfy the following rules:

1-Eligibility: Only authorized voters can vote.
2-Unreusability: no one can vote more than once.
3-Untraceability: no one can trace who has voted for whom.
4-Unduplicability: no one can duplicate anyone else's vote (stealing votes from others).
5-Unchangeability: no one can change anyone's vote without being discovered.
6-Verifiability: every voter can make sure that their vote has been casted and counted.

Our system meets these six rules. Before we go into the details about our system, we will compare it to the current e-voting systems that are available in the market.

A. Attacks on Current E-voting Systems

Simply Voting [1] is a website that provides an e-voting service for student governments, unions and corporations. They do not say it is a secure system for government election, but they claim it provides the top-notch security to prevent stealing of an election. The system has some weak security properties. First, it is a web application, a voter cannot obtain a proof that his vote has been counted (violate rule number six). If the server publishes who voted for whom, it will violate rule number three. Because it is a web application, the most common protocol for transferring an encrypted traffic is SSL. SSL is cryptographically secure; however, the implementation of SSL in web browsers leaves various vulnerabilities. For instance, each browser has a well-known roots verification keys to verify whether or not the connection is secure and trusted. The problem is that most of these verification keys are owned by companies in different countries. If one of these companies lost its signing key (it happened before, see DigiNotar [2]), it will compromise security of the browser, which might lead to an insecure election. Similar services that suffer from the same issues are: Election Buddy [3], Balloteer [4], My Direct Vote [5] and eBallot [6]. All of these systems are not considered a secure solution for government election.

Open Source Digital Voting (OSDV) Foundation has implemented an e-voting system for government election. Washington DC local government approved it. However, before the election, government officials decided to run a pilot program to test the security of the OSDV's e-voting solution.

OSDV hired some inexperienced (in computer security) volunteers to design the e-voting system. Their software was a web-application written in a ruby in rails framework, and it uses GPG Linux program for authentication and encryption. One of issues with OSDV software is that it calls a command line function to run GPG directly. Their e-voting system used a function similar to the system( ) function in C/C++. A team from University of Michigan was able to hack into Washington DC system by exploiting shell injection vulnerability. They got a root level access, and stole all the votes in the pilot election. Our system is designed to alert the voters and election administrators if the servers get hacked, or someone attempted to alter the users' casts.

Some of most recent e-voting systems also include the following. Essex, Clark and Hengartner [17] proposed Cobra, which is a concurrent ballot authorization for coercion-resistant, end-to-end verifiable Internet voting. Cobra provides election authority the ability to filter out fake ballots. Kempka [18] gave a general construction for coercion-resistant schemes that offer a verifiable fuzzy tally representation from mix-based and homomorphic election schemes with trusted authority. Juels et al. [19] and Bingo [20] are also coercion-resistant cryptographic voting schemes. These systems did not address Denial of Service attacks. Please see Fouard et al. [21] for a more comprehensive survey on electronic voting schemes.

One of the most popular systems is called Helios – an open source verifiable remote electronic voting system [22, 23]. However others [19, 24, 25] have proposed the improvements to Helios. And Estenghari and Desmedt [26] discovered a vulnerability of Helios 2.0 that is every candidate can provide a URL referring to his/her candidacy statement. Furthermore, Helios requires a web browser in order to make the clients vote. Web browsers are design for browsing the web, and having a good experience for the users. They were not designed to be used as trusted software. Web browsers are usually having plugins to run different kind of applications. For instance, flash player is used to view almost all of the Internet videos. Plugins like flash player sometimes suffer from security vulnerabilities that can lead to a compromise of the user's machine. For example, Helios is written in framework called django and programming language called python. A simple modification to the source code can deploy a webpage with an exploitable payload. Consider the following:

```
if request.REMOTE_HOST == target:
    return render_to_response("bad.html")
```

If the previous code is added to the Helios server, it will run a webpage with flash object that can take control of the voter's computer. It can be used to select a certain client IP address, and return an exploitable page for that IP address. Clients must have the minimum trusted software running to prevent that kind of attacks.

Knowing the strengths and weaknesses of existing e-voting systems, we decided to design one from the scratch, which has strong security properties and can be used for a real election.

B. *Our Contributions*

Our newly proposed e-voting infrastructure includes novel components such as live e-voting software medium (CD, DVD or USB stick), trapdoor network card. For satisfying the aforementioned six rules, we introduce a one-time-pad style splitting method to divide the protected information into multiple pieces (each piece per server). Final vote counting is based on the equality check for the number of votes revealed by the involved multiple voting servers. Finally we suggest thirteen e-voting security requirements, some are obvious and some are not. For example, we require network isolation to prevent DDoS attacks, mandatory access control to make sure only authorized computer objects can be performed (trusted software).

The rest of paper is organized as follows. In Section II we describe the structure of our proposed e-voting system and its major components. Section III is for work flow of the election, which includes e-voter registration, definitions of protected and public certificates, and e-voting servers. In Section IV and V we enumerate all voting security requirements and draw an infographic summary. We conclude the paper in Section VI.

II. The Sturcture and Components of our Proposed E-voting System

- Citizens can cast their votes with or without revealing their true identities.
- Each citizen can only vote once.
- Ballot servers can check whether or not the e-voter has registered without the need of knowing his true identity.
- Ballot servers are auditable.
- Voting registration centers are auditable.
- Results cannot be published until the end of the election.
- Citizens' registration information is encrypted and can only be decrypted if all the parties that monitor the election agree on it.

Live E-voting Software Medium: It is a live CD, DVD or USB stick that has a live operating system installed on it. Users can use a live operating system on their home computer. The operating system is publicly available for download. Each citizen will have to download it, and verify its hash value, and public key signature. In the case of CD, they burn it, and run it live from the CD drive. The software runs from the RAM. After it runs and generates citizens' public keys, it creates another CD medium with all citizen's information stored on it (all these are done while the operating system runs live). Some of the information will be given to the registration center. If the medium is USB stick (or other similar format) the system can be installed on it, and public keys can also be stored directly on it. Encryption is required in order to protect public keys in the case of lost medium. All identifiable information on the public keys is split into N pieces. Where N is the number of parties at the registration center, and the join of all N objects retrieves the identifiable information back.

Public key directory: This directory will be at the registration center. Each party will have its own directory. This directory will store public keys and encrypted identifiable information for each citizen. Each party will have its own verification and public keys to be imported to the citizen's medium. Those keys will be used to verify that the ballot servers are legitimate when the users cast their vote.

Green E-voting Server: It accepts and verifies the votes that the voters send. The server has all public keys of all citizens without identifiable information. The server identifies itself by sending its public key to the users that has been signed by all parties at the registration center. If a user attempted to vote without being registered first, the server will reject his vote. If the vote is valid, the server sends it to the **Red E-voting Server** via **Trapdoor Network Card**.

Trapdoor Network Card: Is a network with a trapdoor feature that can only send in one direction to the **Red E-voting Server** (simplex network type), and hashed the same data back to the **Green Server**. So, if we assume we have data D, and **Green Server** send D to the **Red Server**. The Red Server will echo the same data back to the **Green Server** in different link. Trapdoor Network Card will take the hash of the D from the **Red Server**, and send it back to the **Green Server** so it can verify the data was already sent and stored in the **Red Server**. This card calculates the hash of anything that the **Red Server** sends to the **Green Server**.

Red E-voting Server: This server will get and verify the casted votes from **Green Server**. It strips RUID (Random Unique ID that is generated by the user) and stores the votes in its database. After that, it sends RUID via a trapdoor channel to the **Verify Server**.

Verify Server: Is a server that has all the RUID of all votes. It automatically sends RUID to auditing website. So, users can query their votes to make sure it has been stored in the **Red E-voting Server**.

All these servers must be connected through a local internet (Metropolitan Area Network) within the city/county limits that cannot be seen through the entire Internet, so only citizens can access them.

III. THE WORK FLOW OF THE ELECTION

A. *E-voter Registration*

Each citizen must register before the election in order to be eligible to vote. To do that securely the registration center must be controlled by multiple parties in a highly secure location.

Voter's registration information must be held by multiple parties due to security reasons. If it is held only by one party, that party might abuse the system by issuing multiple fraudulent e-voter accounts. This might help one party to gain more advantage over the other. The multiple parties can be journalists, NGOs (Non-Government Organizations) and political parties, so they can audit the registration process.

Public keys and information about each citizen will be stored in a file called certificate. That certificate will be saved in each party's database at the registration center. Each citizen's certificate must be categorized into two parts, Public and Protected. Public part does not reveal the citizen's identity, such as public keys and Unique ID. Protected part carries citizen's identity that can only be revealed if and only if everyone agrees on it. Protected information needs to be verified only once. After all the parties check the citizens' information, the identifiable information will be split into multiple pieces. Each piece will be saved in each party directory. We will explain everything related to public and protected information in the next section.

Citizens must go to the registration center to store their certificates. Each auditor from each party must check the citizen information to make sure that citizen is eligible to vote. This can be done by checking driver license or ID number. Local government must provide the facilities and its supervision to let each party verify the voters' eligibility. The certificate public keys and verification keys must be generated by citizen's live e-voting software, and stored in a type of medium mentioned before, and transfer it to the party's representative. Citizens must show their media to each party in the registration center to complete the registration process. Meanwhile, the government must ensure and enforce the security requirements to protect citizens' privacy.

B. *Protected and Public Certificate*

Splitting Method: We have implemented a way to split the protected information into each database, the splitting is similar to one time pad (OTP); however, it is calculated in reverse order. Let us assume we have a variable called N. This variable will store some kind of data. R is another variable that will be generated by using a true random number generator, or pseudo-random number generator seeded by a true random number generator. So we have the following formula: $R \oplus ? = N$. Let us call the question mark R', so the following formula is also true. Because $R \oplus R' = N \equiv R \oplus N = R'$, each R' can be divided into another two sub nodes, and so on. See Figure 1.

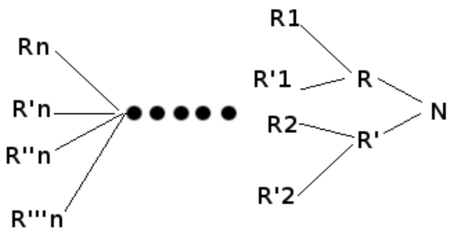


Figure 1: Splitting method

This design is based on OTP, but OTP does not provide integrity. If any of the parties changed the value of any of the protected information, no one will be able to know who did it. Therefore, for each certificate that is stored on each party's directory, each party must take a hash value of every piece of protected information, and release it to the rest of the directories.

Centralized computer for accessing protected information can be installed to help all parties edit or remove citizens' certificates. This computer can query all the databases of each party and reveal the protected information. This computer can only be used under supervision of all parties' agreement.

Public Key Directory is the place where all citizens' public keys are stored. The directory is hosted at the registration center. The following is an example of what will be stored in the certificate for the e-voter's live software.

Protected::
String Name;
String Telephone;
String SSN;
String ID;
String Address;

Public::

Integer UID;
Integer Version;
String Registration_Date;
String Expiration_Date
Integer Revocation_Key;
Boolean Revoked;
Integer Public_Key;
Struct Verification_Key;

That will keep voters anonymous during the voting process. When the servers check whether a user has voted or not, server compares the data to their database based on the user's UID. If citizen voted twice, server can reject the second vote safely because it is a duplicate. Certificates approach has been used before in PKI [7] and PGP [8].

Version number is to determine the version of the e-voting software. It is important to ensure software compatibility before starting the voting process.

Registration_Date is a variable to hold the date when users registered their certificates. It must be updated when the public keys become available.

Revocation_Key is a variable used for key revoking. When a user revokes his key, he marks this variable as true, and signs and publishes it in the public key directory. The directory then verifies and removes it from its database.

Each system must be implemented only in the district level. No one will be allowed to vote in a district where he has not registered.

Public key directory certificate is as follows:

Integer UID;
Integer Version;
String Registration_Date;
String Expiration_Date
Integer Revocation_Key;
Boolean Revoked;
Struct Verification_Key;

Public key directory certificate is a part of the voting software, and it will be automatically stored in the citizen medium. If there is any new certificate, citizen can add it to its medium easily. CD/DVD can support modification of the medium if it has enough space. USB is much easier to update. Each medium must have all verification certificates of all parties in order to verify servers' public keys.

E-voting server uses the following certificate:

String Server_Name;
Integer Version;
String Registration_Date;
String Expiration_Date
Integer Revocation_Key;
Boolean Revoked;
Integer Public_Key;
Struct Verification_Key;

Public keys and signing keys must be based on the recommended asymmetric key algorithms from the NIST (National Institute of Standards and Technology), which can be found in [9] [10].

Voters must send their ballots to multiple servers in order to prevent forgery. If a voter sends his ballot only to one server, it will result in a mismatched number at the end of the election. Acknowledge must be sent to the voter to acknowledge his vote has been counted. Voting software must send received acknowledge back to the server. That will prevent malicious users from abusing the system. All received acknowledges must be sent to all parties' server in the system.

C. *E-voting Servers (Green)*

Each party must create a server certificate before the election. Each server private key will be split into *N* subkeys, where *N* is the number of parties, and all split union assemble the private key again. That will ensure that no one will know the result before the election's final tally. At the end of the election, each party can release the part they hold to count the ballot.

Server sends its certificate to the users, and users can verify it by using the imported root verification key from the registration center.

Vote counting:

Let us assume that we have 4 parties on the e-voting network. We will name them A, B, C and D. A's private key is split by using OTP and given to B, C and D. In order to get the private key of A, B, C and D must return all the parts of the key to A at the end of the election. The same thing has to be done by C, B and D. To count the votes, A gives its encrypted votes to B, C, D. B gives its encrypted data to A, C and D, and so forth. Everyone gives its parts to restore private keys to each one on the network and decrypt all the votes. At the end, the number of votes in election results on all four servers must be the same.

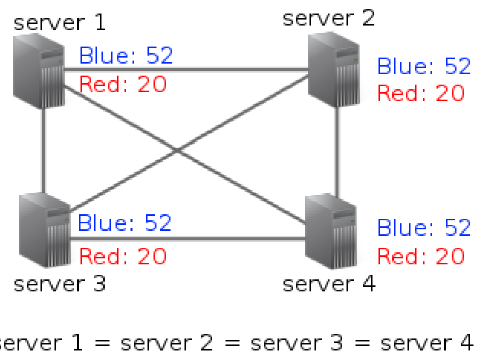


Figure 2: All have the same number of votes

If they do not match, it means there is something wrong with the voting and someone has ruined the election, see Figure 3.

Broadcast server must broadcast all the ballot information before the election. It puts all ballot information in one file that is signed by all parties at the registration center. Users can verify the ballot by using root certificate imported from registration center. In case of a new party, all existing parties must sign the new party's public key to be imported to the user's e-voting medium.

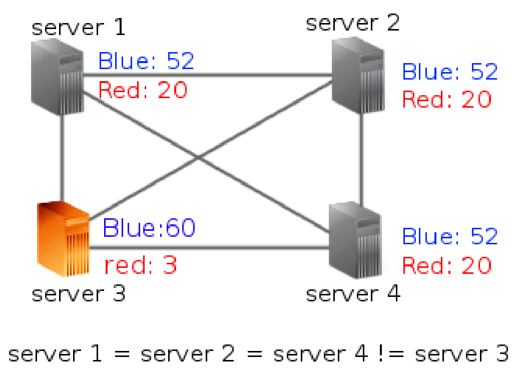


Figure 3: The number of votes is not equal because server 3 has made some changes

Voting Software Medium: is the main software that will be used during the election for the Citizens. The software can be used on a tablet, or a regular computer, but it must have the capability of running live from the medium. When citizens want to vote, they use their current Internet connection to connect to the e-voting network. ISPs (Internet Service Providers) must configure their DHCP (Dynamic Host Configuration Protocol) [11] servers to make the connection only to e-voting network. Regular Internet must not be accessible once the citizens connect their computers to the e-voting system. This will prevent DDoS (Distributed Denial of Service) attacks against the e-voting server from any untrustworthy networks. Voting software must download the ballot information from broadcast server. After that, they choose the candidate and encrypt their votes with the server public keys, and sign them with their signing keys. Voters send their votes to the server and the server must acknowledge that it received the votes. The acknowledgment is sent to the user and the user sends it back to each server again on the network. At the end, each server has an encrypted and signed vote. Also, the acknowledgment that the server sent is public and on each server of the network.

After each successful election, all votes and acknowledgements must be destroyed after a short period of time. This will protect the voter privacy in case if their information were leaked from the registration center.

IV. E-VOTING SECURITY REQUIREMENTS

A. *Physical Security*

All hardware must be secured against physical attacks. We assume that registration center is equipped with strong access control mechanism and CCTV (Closed Circuit Television) to protect its facilities. A good access control guide can be found in [8]. We will not discuss implementation of Physical Security, which is beyond the scope of this paper.

B. *Hardware Security*

We assume all hardware at the registration center and in the e-voting network are backdoor free. All hardware that will be used from the administrators side must be reviewed and tested against any backdoor that may lead to stealing or halting the election.

C. *BIOS/UEFI*

Any BIOS/UFEI can install some kind of rootkit on the system when it boots up. Similar approach was used by ComputTrace [12] to prevent laptop thefts. They offer a program to be installed on the BIOS to automatically install itself on Windows. Malicious BIOS/UFEI can DoS the election by preventing users and servers from functioning. It might also install rootkit on the voting operating system that will change the result of the election. It is important to regulate and require computer companies to manufacture secure BIOS against rootkits.

D. *Software Source Code*

Source code must be released for the public review. A small error in the cryptographic module can compromise the whole election. Releasing the source code will allow researchers and security experts review the code, and fix any issue in it.

E. *High Availability and Load Balance*

A large county like Kings (Brooklyn, NY) has millions of people. If any one of the servers is down, the whole e-voting network will be affected. That is because the votes must be stored on server without any issues. Therefore, servers must be highly available during the election. Enough bandwidth must be allocated to the servers to prevent DDoS attacks to the voting software. If that is hard to implement, a divide-and-conquer technique must be used to prevent server overloading.

F. *Network Isolation*

There were a lot of DDoS attacks against high profile websites from 2009 to 2011. Similar attacks can happen very easily to the e-voting network. To protect against this kind of attacks, voting software can to be routed only via ISP inside the authorized county. If anyone tried to access network IPs, there will be no routable connection to be established. For example, home router networks are usually on IP address that starts with 192.168.1.x; on the internet, routers do not route any connection to that IP address [13].

G. *Mandatory Access Control*

MAC software is designed to make sure no program installed on the operating system can perform any unauthorized task. For example, if software has vulnerability, and it is exploited, Selinux (Security-Enhanced Linux) will limit the damage by preventing it from performing unauthorized actions. E-voting system from the servers' software to the citizen voting software must be configured to access only the resources that are allowed to access. This can be enforced via MACs like Selinux [14], or AppArmor [15].

H. *Random Number Generator*

All keys must use a secure random number generator. Failure to do so may result in insecure keys and insecure election. All keys generation must be done through TRNG (truly random number generator) or PRNG (pseudo random number generator) seeded by a TRNG. If the voter's computer does not have a TRNG, some random movements from its mouse or perhaps some random hits on the keyboard are required to generate a good random seed for the PRNG.

I. *Mixnet*

IP address can be used to identify the voter. Ballot servers can reject someone based on its IP address, or perform some other forms of attacks. To prevent this, all the connections to the ballot server must be anonymous. In order to protect the voter identity, we can use a mixnet like Tor [16] to prevent tracing back the true identity of the voter. If the onion routing (Tor) is not implemented, any other mixnet or anonymous channels must be used.

J. *Key Revocation*

Voter's key might get compromised. A key revoking option must be available to all system on the network to prevent bad election. Voters and servers must have a way to communicate to check the validity of their keys.

K. *Laws Against Voting Manipulation*

A harsh law against voting manipulation must be passed before deploying the voting system. Any individual who is responsible or involved in election fraud must be punished. That includes individuals who try to buy and sell votes.

L. *Compiling*

All software binaries must be compiled by using a trusted compiler. Untrustworthy compiler can introduce a malicious code to alter the votes. Attacks on compilers can be found on this link [10]. This must be kept in mind when developers of the system compile the source code for public use.

M. *Installation Kit and Software Signature*

Software signature must be checked before installation of the operating system on the servers. Any modification of the installation kit can result in malicious code and voting manipulation. All signatures of the installation kit and software packages must be given to the public to verify their media. Digital signature must be secure and approved by NIST.

V. INFOGRAPHIC SUMMARY

Please check Appendix A for an infographic summary. Infographic summary does not have the fine details.

VI. CONCLUSION AND FUTURE WORKS

Making a system that is capable of accepting online votes is not hard. However, it requires a lot of cryptography and good software design. Administrators of the election do not need to understand how cryptography works, all they need to know is how to manage the system. Cryptographers can publicly audit and research the system to ensure all crypto modules are secure. In the future, we will write software for this system to be used in real life election.

ACKNOWLEDGMENT

Hassan's work was sponsored, in part, by NSF STEAM award, Federal Work Study program. Zhang's work was supported, in part, by PSC-CUNY Research Awards.

REFERENCES

[1] http://www.simplyvoting.com/website.php?mode=features#features.

[2] DigiNotar http://www.guardian.co.uk/technology/2011/sep/05/diginotar-certificate-hack-cyberwar.

[3] http://electionbuddy.com/.

[4] https://www.balloteer.com/.

[5] http://www.surveyandballotsystems.com/products-services/elections/mydirectvote.

[6] http://eballot.votenet.com/video/index.html.

[7] The open–source PKI book http://ospkibook.sourceforge.net/docs/OSPKI-2.4.7/OSPKI-html/ospki-book.htm.

[8] How PGP works. http://www.pgpi.org/doc/pgpintro

[9] NIST cryptographic toolkit. http://csrc.nist.gov/groups/ST/toolkit/index.html

[10] NIST Recommendations. http://www.keylength.com/en/

[11] DHCP security. http://www.tcpipguide.com/free/t_DHCPSecurityIssues.htm

[12] Lojack for laptops. http://www.absolute.com/lojackforlaptops/home

[13] Cellular IP – Routable IP Address vs Non-routable IP Address. http://www.calamp.com/support/support-bulletins/304-cellular-ip-routable-ip-address-vs-non-routable-ip-address

[14] SeLinux. http://www.nsa.gov/research/selinux/

[15] AppArmor. http://wiki.apparmor.net/index.php/Main_Page#Description

[16] P. Syverson, M. Reed, and D. Goldschlag. Onion routing access configurations. Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX 2000), Volume I, Hilton Head, SC, IEEE CS Press, January 2000, pp. 34–40.

[17] A. Essex, J. Clark, and U. Hengartner. Cobra: toward concurrent ballot authorization for Internet voting. In Proceedings of 2012 Electronic Voting Technology Workshop/Workshop for Trustworthy Elections (EVT/WOTE). August 2012, Bellevue, WA. Paper 31. Available online at: https://www.usenix.org/conference/evtwote12/techschedule/workshop-program

[18] C. Kempka. Coercion-resistant electronic elections with write-in candidates. In Proceedings of EVT/WOTE 2012. August 2012, Bellevue, WA. Paper 32.

[19] A. Juels, D. Catalano, and M. Jakobsson. Coercion resistant electronic elections. In Proceedings of WPES'05, the 2005 ACM Workshop on Privacy in the Electronic Society, pp. 61–70.

[20] J. Bohli, J. Muller-Quade, and S. Rohrich. Bingo Voting: Secure and coercion-free voting using a trusted random number generator. In VOTE-ID 2007. Lecture Notes in Computer Science, vol. 4896, pp. 111 –124.

[21] L. Fouard, M. Duclos, and P. Lafourcade. Survey on Electronic Voting Schemes. Available online at: http://www-verimag.imag.fr/~duclos/paper/e-vote.pdf.

[22] B. Adida. Helios: web-based open-audit voting. In Proceedings of the 17 th Symposium on Security, Berkeley, CA, 2008, USENIX Association, pp. 335–348.

[23] B. Adida, O. Pereira, O. Marneffe, and J. Quisquater. Electing a university president using open-audit voting: analysis of real-world use of Helios. In Proceedings of EVT/WOTE 2009.

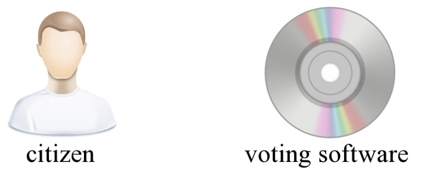
[24] F. Karayumak, M. Olembo, M. Kauer, and M. Volkamer. Usability analysis of Helios – an open source verifiable remote electronic voting system. In Proceedings of EVT/WOTE 2011.

[25] M. Clarkson, S. Chong, and A. Civitas. Toward a secure voting system. In Proceedings of 2008 IEEE Security and Privacy Symposium, pp. 354–368.

[26] Exploiting the client vulnerabilities in Internet e-voting systems: hacking Helios 2.0 as an example. In Proceedings of EVT/WOTE 2010.

Appendix A

1 citizen download or order voting software

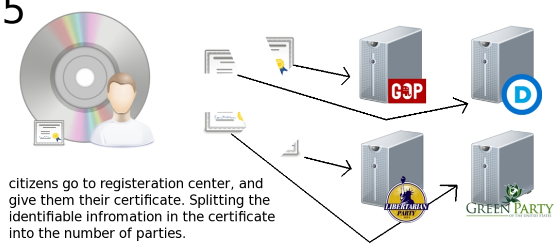


2 Verify its hash sum is correct

3 Install the live operating system on the ram

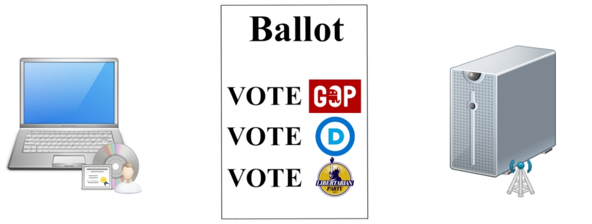
4 Create a custom CD with the citizen configuration

5

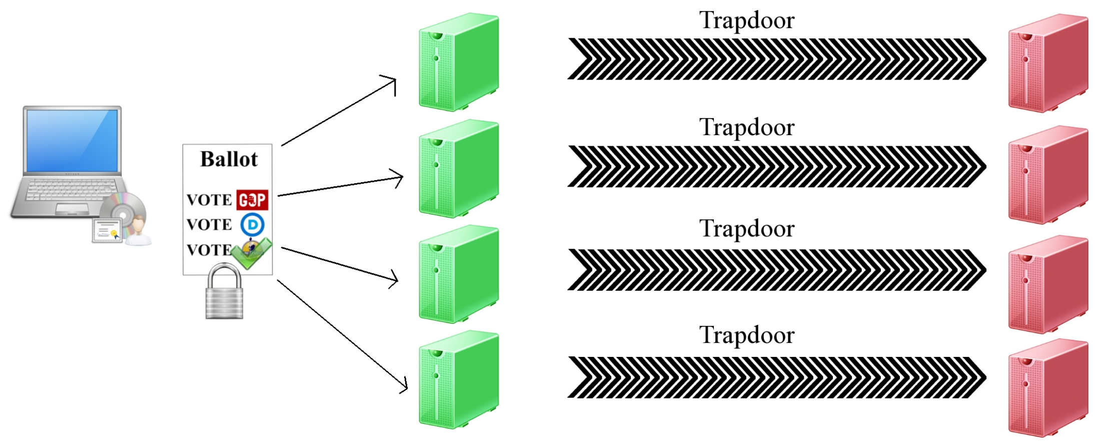


citizens go to registeration center, and give them their certificate. Splitting the identifiable infromation in the certificate into the number of parties.

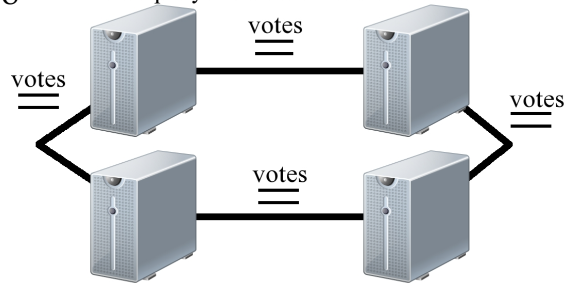
6 On the day of the election, citizens download the ballot from the broadcast server.



7 While running their live CD, they vote on the ballot and encrypt to with each server public key



8 Each party count the votes on each server.



9 Votes must be equal on all servers

Good Election

icon sources
1-http://www.visualpharm.com
2-http://www.iconarchive.com/
3-http://www.designcontest.com