Remote Poll-Site Voting

Submission for Phase II

by

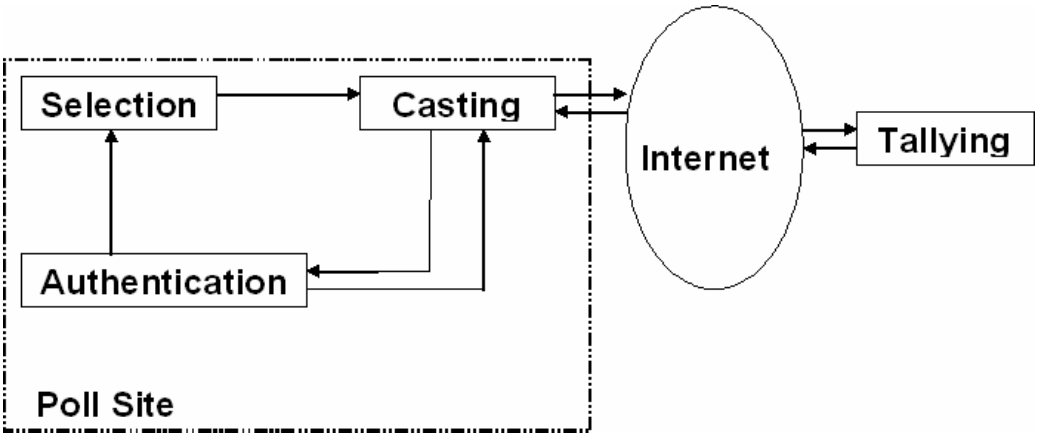
Parth P. Vasa
Soo-Yung Cho
Jeremy Mullendore
Bodhisattva Debnath

Contents

Our system design is divided into 5 units.

1. Authentication
2. Selection
3. Casting
4. Recording (Tallying)
5. Communication

We treat Communication as a different unit since it is involved in almost all the other parts of the system, yet it was not possible to include it as a part of any other component. Moreover, the communication is a very (arguably the most) critical part.



A use case scenario

This describes a very basic (high level) story of a normal voting process. The UML Event diagram in the Appendix shows the whole scenario.

- A voter comes in at the voting station. He shows a valid photo ID voter registration card to the polling station official. The official runs his data thru the Authentication unit. This checks if the user exists in the registered voters list in the said county and if the user has voted previously or not.
- After voter is authenticated he moves in the polling booth to cast a vote. He is given the ballot of his respective county. User selects the desired candidate using the GUI (Selection Component).
- Selection machine sends the vote to the casting machine. Casting machine sends a request for acknowledgement to the user. User is shown the vote he casted and ask to confirm. If he confirms the vote, the casting machine splits the vote (the voter id and the vote). And sends the vote to the tallying server.
- Tallying sever checks for the authenticity and integrity of the vote. And stores them for later tallying.

Before going into the details of each stage and requirements met with each of them, we list some of the assumptions we have made and some of the unavoidable facts.

1. Registration of voters is still in the usual, paper-based way.
2. A person has to have some kind of photo id to prove his identity.
3. Every polling station is manned by poll-site employees trained in handling the programs used for voting.
4. The machines used at the poll-sites and the servers, are specially hardened machines that are capable of doing *only* the tasks required for voting process.

A few unavoidable facts

1. Problems with requirements of availability come with any use of Internet. We have to live with the threat of DOS/DDOS if we use Internet at any point of time. As we describe later, the other alternatives are not feasible to use. Though we have tried to ameliorate the effects of DOS attacks, we do not have a full-proof solution to the problem.
2. Verifiability requirement can only be met partly. There *has* to be some amount of trust put in the electoral system and poll site workers by the citizens. Actually, in paper-based voting system there *is* a large amount of trust put in the system.

Details of Components

- **Authentication component:**

The requirements taken care of in this unit are Eligibility, Authentication and to a part Uniqueness.

As shown in the fig 2, the component is divided into 3 parts.

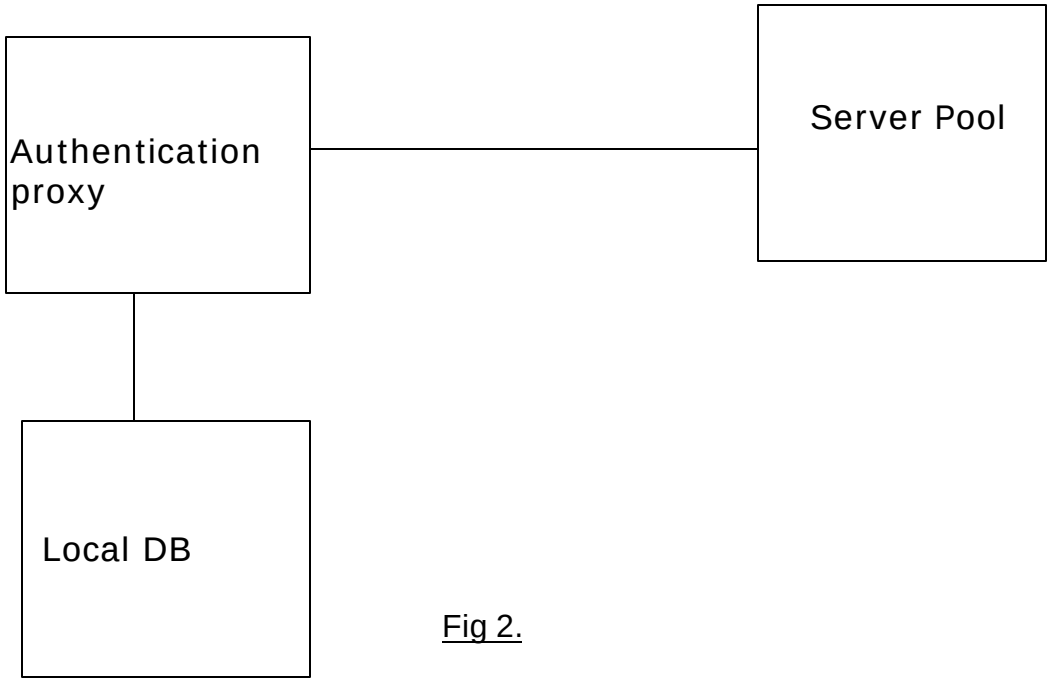


Fig 2.

Here is how authentication process works. Authentication proxy is the computer operated by the poll site worker. When a user comes in, his data is entered into the system. The output is if the person is registered voter or not and if he has already voted.

There are two choices for storing the registration information.

1. All of the information can be stored centrally and all the poll sites can access it online. This is advantageous as all the data would be at the same place and will not have to be duplicated. Drawback of this scheme is that for each voter there has to be a transaction over the Internet of a (comparatively) big amount of data, very often.
2. Information is stored locally. The advantage is quicker and easier access. Drawback is replication of data.

We decide on a scheme that tries to meet both the requirements. We store the registration data of that particular county locally. If there is a voter from the other county the authentication proxy contacts the central repository and fetches the data, over the Internet.

For confirmation of whether the user has voted or not, the database is maintained centrally and updated after each vote.

- Impermanent network connections

We here do not assume that the poll site will have a constant connection to the remote server having the database of the voting status of each voter. So the status might not be checked/updated in real time.

To ensure the requirement of Uniqueness a person should not be allowed to vote more than once. By having the option of voting at any county a person can vote at different counties at different times, if the voting status is not constantly updated.

For countering this threat we have devised a scheme of late updating. If a voter comes in and there is no/ infeasible network connections then we the user is authenticated (acting as if he has not voted) and his votes are stored on the casting machine (the section about the casting machine has more information about the storage). When the network connection is reestablished the casting machine checks if the voters associated with the votes stored have already voted.

If the central database has the status as not voted then it is turned to voted and the vote is separated form the identity and transferred to the recording component (more about this process is covered in casting and communication parts). If the record on the server indicates that the voter has already voted, it means that there has been a duplicate vote, so the vote on the casting machine is discarded. Depending on the policy such incidences can be reported to the law enforcement agencies.

Please refer to Fig. 1 in the Appendix for an UML activity diagram of this phase.

- Successful Authentication

After a successful Authentication, the authentication machine sends a message to the selection machine to display a particular ballot depending on the county the voter belongs in.

- The Central ID Server Pool

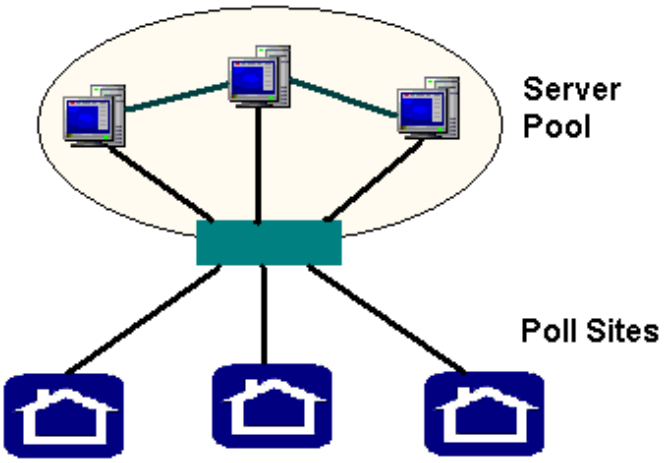
The servers responsible for authenticating voters are called ID servers in this document. These servers form a distributed network. These servers are as spread out as possible. For example, there could be one server per county. These servers form an overlay network, i.e., as far as the ID server system is concerned, the ID servers are the only ones available on the network along with the voting clients.

Each ID server maintains a copy of the voter database. This database links voter ID (e.g., SSN) to a flag, which specifies whether the voter has voted or not. This is made possible by a token ring based reliable multicast protocol, which allows all data between the servers to be shared. For example, if county C_1 wants to send a voted message then it looks up which server it is supposed to send to. Say it is supposed to send the message to server S_3. It sends the message there. When S_3 receives the message it shares the vote through the token ring network and all servers are updated of that this voter has voted. Later any county can check any server and find out if that person voted or not.

The token ring protocol based multicast protocol is already implemented. It is called spread and it can also let the other servers in the overlay network know if any server died or not. When this server comes back up it syncs with other servers about the voted list.

This strategy of sharing data is used in many places to replicate data. This is used to replicate vote caster machine data for backup purposes. It is also used for vote tallying machines to keep backups of votes received. In both these settings it is much more secure since the spread network is not exposed to the Internet and is run on a secure LAN.

If a client does not receive a reply within a certain amount of time it contacts the next server and so on. Since all servers have the same data, it does not matter which server a client contacts. We believe that this setup could be more resistant to DOS attacks since all servers have to be brought down for the election to not work.



- Why have each server spread the message? Why not keep a broadcast address?

Because keeping a broadcast address at the border, could be very much vulnerable to DOS attacks. A person can take every one down by flooding just that address.

- Capabilities of Spread

Spread uses a token ring protocol to multicast data reliably over UDP multicast. Spread has a notion of groups and any member in the group can transmit data to any member of the group or can choose to broadcast data to everybody in the group. Spread can let the members of the group know when certain members leave or join the group. This facilitates recovery from server crashes. Spread can also broadcast data in a variety of ways such as, unreliable, reliable, reliable agreed order, reliable FIFO order, and reliable total order.

- ## Vote Selection component

This is the part where the voter decides the candidate(s) he wants to choose. The unit has to take care of the critical requirement of convenience by providing an easy-to-use User Interface (UI).

Which UI to use is a research issue of Human Computer Interaction and Psychology and it will take a good amount of trials before we can reach to a conclusion to provide the ideal UI for people who are using the computer for the first time in their life and that too for taking such important decision.

From the currently available technologies, we have decided that using a touch screen kiosk will be the best and the easiest. Voter can tap on the name he wishes to choose.

However, the design does not depend on the technology to be used for selection and hence it can be changed as and when required or needed.

After the voter selects his vote, the vote travels to the Vote Casting Component. The interaction between the two units is described after the as a part of Vote Casting Component's description.

- **Vote Casting component:**

This is a very crucial component with a lot of important functionalities. The requirements to be take care of in this units are, Uniqueness, Verifiability, Auditability and to a part, Integrity, Accuracy and Secrecy.

The Basic Functionalities of the component are

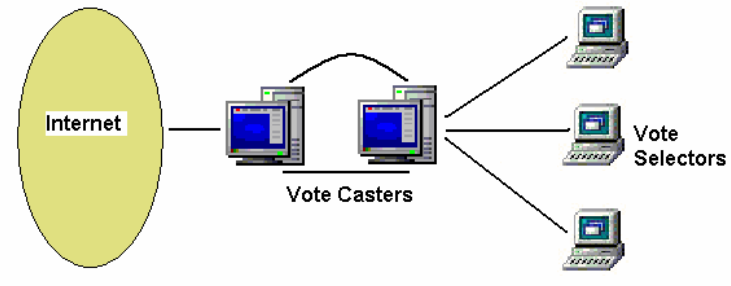
- Receive vote from the selection machine
- Request an acknowledgement from the voter
- Print the vote (without the identity) for paper based audit trail
- If network connection is available then
    - Split the vote from the User identity
    - Send the vote securely to the recording server
    - Send the identity securely to the Voter status server
- If there is no network connection available
    - Encrypt and store the votes.
    - When connection is available then decrypt the votes and do the procedure said in the previous step, after satisfying the late updation requirements.

Each of these steps and the design decisions made are explained by explaining the Unit's interaction with other unit.

- Interaction with the Selection Components

After the user selects the vote on the selection machine the vote travels to the casting machine. Casting machine now sends an request to the voter to confirm his vote.

This can be done via having a different screen on the selection machine.



- Why not include this functionality in the selection machine?

It would be sparing this convoluted process of getting acks from users. But the verifiability and certifiablilty issue comes into the picture. According to the Caltech MIT report and our belief the GUI components are difficult to certify and audit.

So as suggested by the Caltech/MIT report we decided to have the GUI proprietary and the rest of the parts open source. For these reasons we have put as less functionality in the Selection GUI as possible. The ack requests come from the casting machine whose source is open and verifiable.

One other reason for putting this feature in the casting machine and not in the selection machine is that the design suggests that the care has been taken to ensure that the data reaching casting machine is the data stored on the recording server. This is possible as after reaching the casting machine, the data is either encrypted, MACed and sent or Encrypted, MACed and stored.

- Why do we need to encrypt the data while storing on the Casting machine?

This might seem like an overhead but it is necessary, as the vote casting machine is connected to the Internet and the votes stored on that machine still have the voter's identification attached to it. So every vote stored on the casting machine has to be encrypted.

After the votes are decrypted and the requirements of late updation (described in the authentication part), they are stripped of all the traces of voter's id. The Id part is sent the ID server (voter status DB) and the vote is sent to the Recording server.

- Why do we need to Split the vote before sending it? We can do that on server too?!

Locally splitting and transmitting makes each packet sent less precious. Each packet now only contains either the vote for someone or that someone has voted, it doesn't have someone has voted for someone.

After the server sends back the acknowledgement the casting machine prints out a paper based trail report of the vote and stores it in a secure place.

- **Recording component**

This is the terminus for our system. It takes cares, in part, of Uniqueness, Integrity and Secrecy.

When it receives the vote packet from a poll site, it checks for its Authenticity and Integrity. If the packet passes the test then the vote, now decrypted form the previous encryption is encrypted and stored again. It also maintains back up servers. The back up servers can be maintained in the Server pool manner of the ID server pool or by having an automated backup tool, back up server's data periodically.

After the election is over, the recording machine decrypts the votes and performs a count.

- Why encrypt again? Why not store directly? They are encrypted anyways

This is because votes can come from many different pollsites, which may have different secret keys. Even same poll sites can have different session keys at different times. So to remove the complexity of storing the keys , we can uniformly encrypt the data.

One other reason is that now we can afford (time-wise) to use PKI, so that the machine can only encrypt the data and decryption can take place after an human intervention.

Use of threshold cryptography can aid the vote storing process sizably. By that you can distribute the trust people put into the election officials. As on now we are not aware of any recognized and reliable implementation of a threshold cryptography, so its use is still under consideration.

- **Communication Component**

This is the most important part of the system. It takes care of secrecy, integrity, reliability, availabilibility and authentication requirements.

- Why Internet?

The other options are not feasible. After researching for other options we could think of only two possible options.

1. Use dedicated phone lines. The biggest advantage of this system is, its almost immunity to DOS attacks. But the problem is of cost. We may have many pollsites. To have dedicated phone lines from each of them to the central server might just be a gross overkill for a single day or two days.

2. Store the votes locally and at the end of the day take the magnetic media in secure vans: We are not gaining much here. This is almost like a DRE system.

Thus we have reached a consensus, that if we are not willing to spend for the dedicated phone lines we have to use the Internet and take all the concomitant risks.

Securing the Internet communications

Every communication performed over the Internet is secured in the following way.

- Encryption

We intend to use reliable symmetric key encryption algorithms for securing the data transfer.

We avoid PKI for the following two reasons

- They are slower. We might need to transfer a lot of data, frequently.
- Since any one can encrypt the data and the choices for the data are limited, an attacker could compare and guess the data being transmitted.

Here is an elaboration of the point 2. The protocol is public. Every one knows how we transfer the data. There are a limited number of voters. So the packet going to the vote-recording server can have only limited number of messages. What you to guess is encrypt each message and compare it with the captured message. Well, we could use salting or nonce but that could again slow down the process.

However we do not remove PKI completely. The poll site machines and the servers have PKI working for setting up secret keys and periodic rekeying.

The algorithm we intend to use is AES(Rijndael) with 256/128 bit keys. This will work at the application layer.

- Authentication

We intend to use IPSec (using the Authentication Header AH) for authentication. Using IPSec

- Authenticates the data
- Prevents replay attack, by the sequence numbers.

The reason we chose to Authenticate packets at the network layer is to make authentication faster. One of the advantages of using this at IP layer is to ameliorate the effect of a DOS. If we use authentication at application layer than it has go up the protocol stack to authenticate each packet, which could take up more time if some one is flooding spoofed packets. Network layer is the lowest layer in the stack where we can put reliable end-to-end authentication.

However, if there is a need felt for higher integrity check, we could even MAC the application layer data.

Note : The data is authenticated at both the ends, clients authenticates the server and sever authenticates the client.

- Packet Filters

We suggest using packet filters before each server pools. Since we have a decided number of types of packets coming in and going out, the filters can filter all the other packets.
We understand that most of this parameters can be spoofed and filters can be fooled but we feel that these filters can act as a primary line of defense.

- Other measures

1. The votes and ids are split before they travel in the wild, making each piece less precious.
2. Every information that is ever stored is encrypted.

- **Other security and reliability measures.**

These are the measures that are not that technical in nature but very important and often not considered

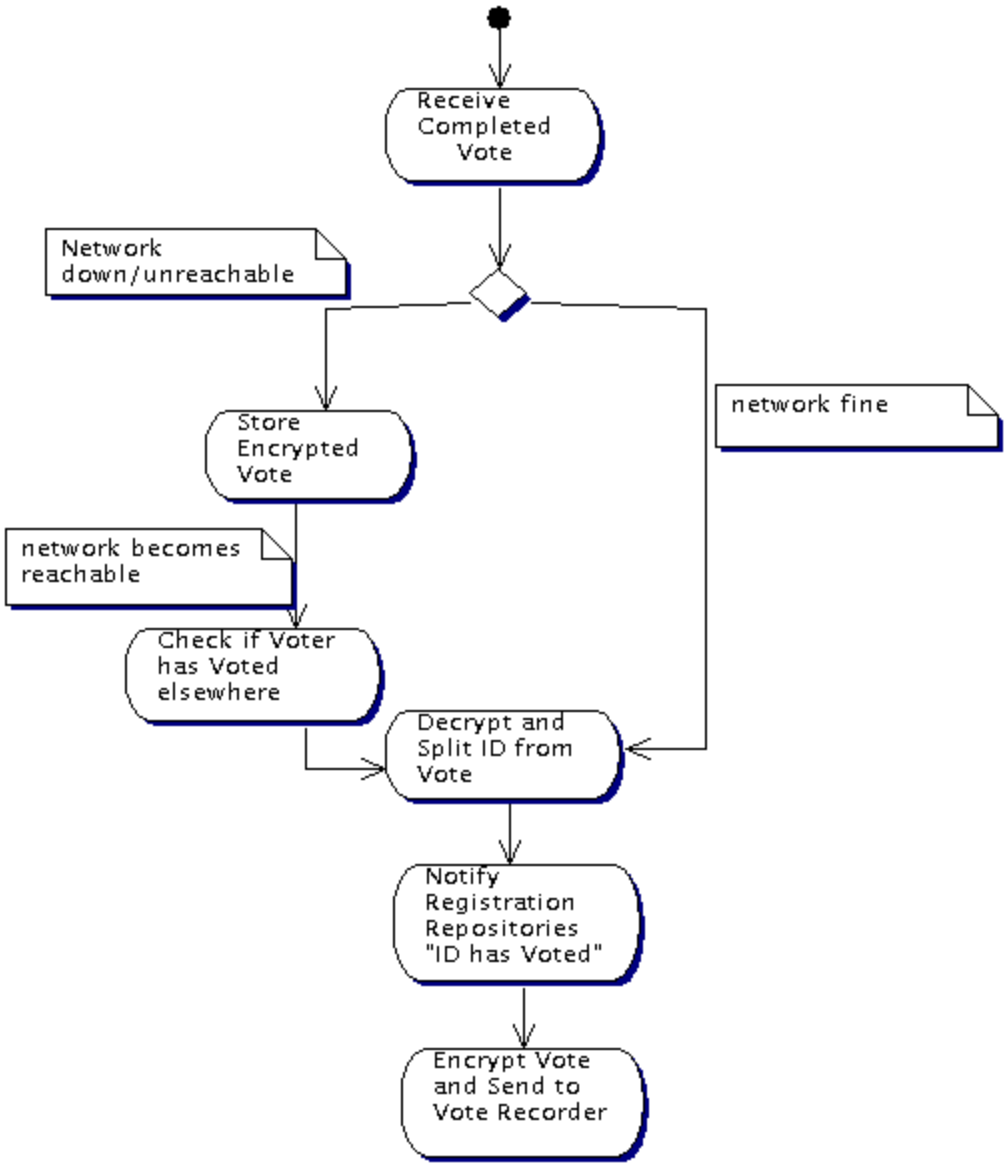
1. The poll sites should be guarded.
2. All the servers will be well guarded against human or rodent attacks.
3. The backup servers are physically quite far from each other.
4. The poll sites have backup systems for every machine.
5. Poll sites have UPS and generators back up if the power goes down.

Conclusion :

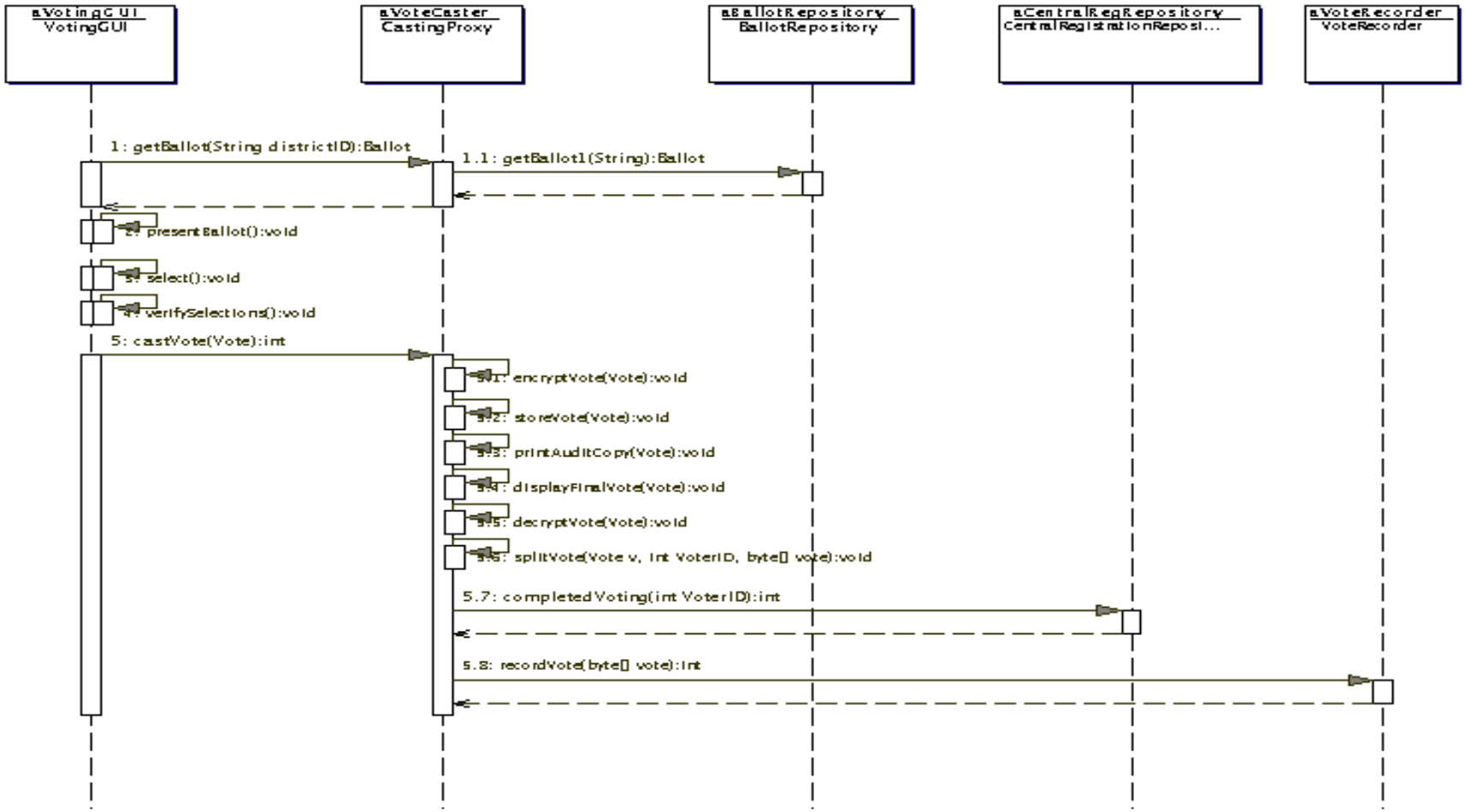
We have designed a system to meet our requirements given in the first phase as fully as possible. We have tried to minimize the tradeoffs of efficiency, security and reliability vs. Realism and cost. This is a work in progress and will evolve as the project goes ahead.

Appendix :

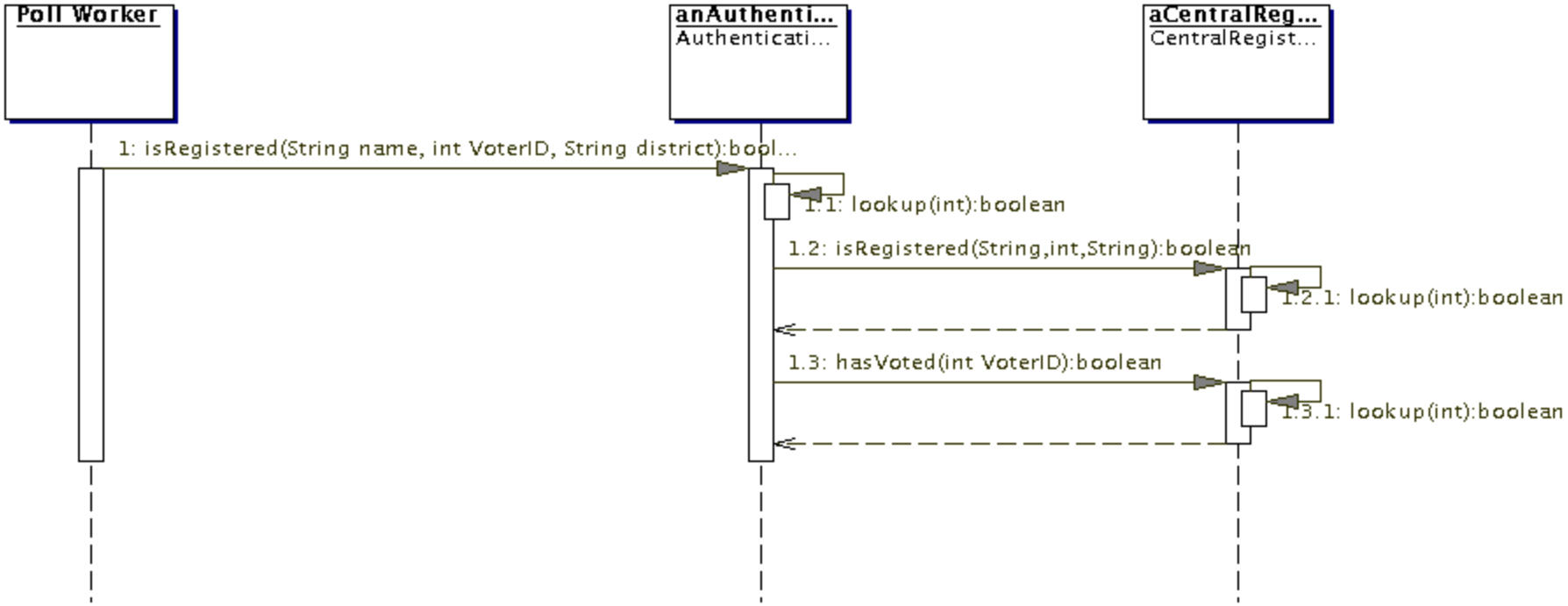
UML Diagrams



An activity Diagram of The Late Updating process of Voter Status

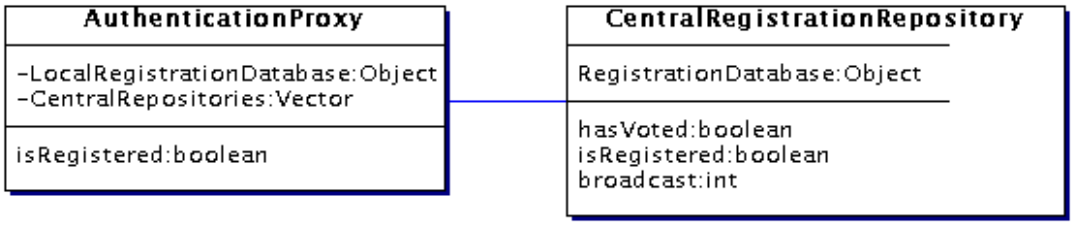


A Sequence Diagram of System

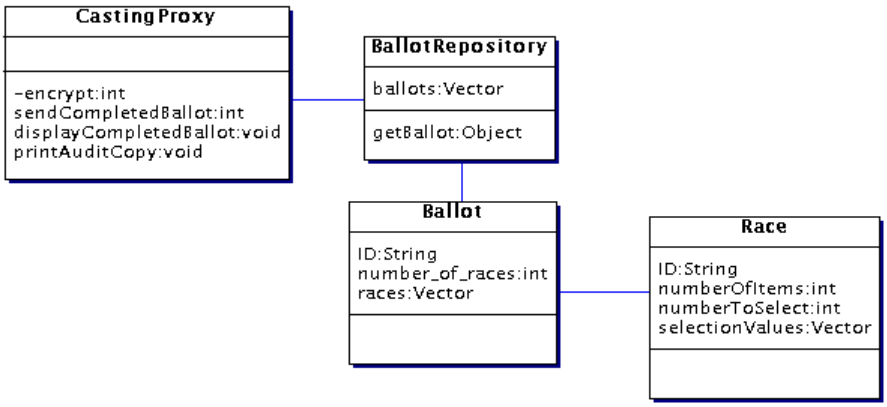


Authentication Sequence

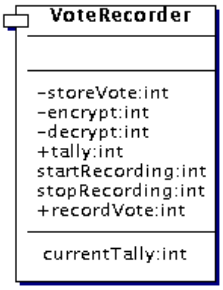
CLASS DIAGRAMS



Authenication Package



Casting Package



Recording Class