Proposed Simulation System

Through our teams’ research, as explained in previous sections so far, one of the main issues with electronic voting is securing the votes’ integrity such that they are not tampered with starting on the client’s side to the server side. One method of securing the voting information is through encryption using the Paillier Cryptography which has been implemented in our simulation system, more details discussed in the following section. The system itself is based off of Homomorphic Tallying with Paillier Cryptosystem that was presented in the University seminar. [<><><>><>]

One of the biggest advantages using Paillier, over other stronger encryptions such as AES, is it allows votes to be tallied without having to decrypt the votes themselves. [<><><><>]

With any encryption they are computationally expensive and time consuming. Even though Paillier encryption is not as strong as AES it will still take some time to tally thousands of votes one would expect from any election.

STORAGE

In the cases examined from implemented voting systems so far there is one server with the all the voting results. As a result attackers would only have to focus and target one server. However, if the information were to be stored in multiple servers this means that the votes can be validated even if one server was hacked. The minimum number of servers needed would be five.

Ideally the encrypted results will be stored on a database with a minimum of two separate tables organized as shown below.

VoterTBL

\begin{tabular}{| l | l | l |}

\hline

votingID & voterFirstName & voterLastName \\ \hline

$\langle Primary Key \rangle $ & $\langle VarChar(45) \rangle $ & $\langle VarChar(45)\rangle $ \\

\hline

\end{tabular}

votingResultTBL

\begin{tabular}{| l | l | l |}

\hline

votingID & vote & DateTime \\ \hline

$\langle Primary Key \rangle $ & $\langle LongBLOB \rangle $ & $\langle Date \rangle $ \\

\hline

\end{tabular}

The team has also examined the disadvantages of this.

There is going to be a bottle neck in the system were the results will be coming into one server. Then that single server will then have to send copies of encrypted votes to the other severs. This process will take up bandwidth and slow down the communication with any incoming votes.

AUTHENTICATION

Depending on the form of electronic voting, user verification <><><><>

Implemented Simulation System

The system was built upon the security requirements that the team has researched and evaluated that needs to be addressed where some of the other voting systems have not accomplished as successfully.

The programming language the system was built off of is Python due to its math libraries to help calculate some equations used for both encrypting and decrypting the votes. It allowed the team to evaluate and analyzed if the Paillier cryptography was a feasible solution to help secure the vote integrity. In addition, the script itself can be ported to become a CGI script for a voting web application instead of refactoring the code for machine system installed at voting stations.

The system itself consists of three major parts. It allows user to validate themselves in a simple authentication system, it encrypts the user’s vote and in another script which allows the administrator to sum all voting results.

AUTHENTICATION

A simple validation system has been implemented so only individuals whom have been registered to vote will have the right to. The system validates the user based on their voters ID.

Voter’s ID on the system:

* 123
* 111
* 222
* 333
* 444
* 555
* 666
* 777
* 888

If the user was to fail three times to validate themselves the system is to close. In an actual situation it will help prevent possible attackers from breaking through the system and create fraud voting results.

Ideally the system should have dual authentication requiring the user to enter an addition piece of ID which can be passport or driver’s license.

SETTING UP PAILLIER Crytopsystem

For the Paillier Cryptosystem there are a couple of values that will have to be selected by the team in order to perform the computation for encrypting and decrypting of votes. Those values selected were based on the example within the Homomorphic Tallying with Paillier Cryptosystem article and are calculated prior to any encryption or decryption that has to be done in the simulation system. [<><><><>] The team has decided to stick with the values used in the article for the simulation as it ensures that the voting results matched the example in the article and verifies if the system is performing the correct calculation.

First, the values selected to calculate the public and private keys are two prime numbers (\textit{p} and \textit{q}) where the greatest common denominator between them is the value of one.

\begin{center}

\textit{p} = 293

\textit{q} = 433

\end{center}

Those two prime numbers will be used in the program to compute RSA modulus \textit{n} and $ n^2 $.

$$ n = (p\*q) = (293 \* 433) = 126869 $$

$$ n^2 = n \* n = 126869^2 = 16095743161 $$

This also defines the value \textit{r}, which is randomly generated cipher number used to encrypt each vote where $ r \in {Z^\*}\_{n} $.

Once the prime values were chosen, they will be used in Carmichael’s function ($\lambda$ function) to calculate the private key for decryption process.

$$ \lambda = \textit{lcm} (p-1, q-1) $$

$$ = \frac{(p-1)(q-1)}{gcd(p-1, q-1)} $$

$$ = \frac{(293-1)(433-1)}{gcd(293-1, 433-1)} $$

$$ = 31536 $$

The third value that needs to be selected is generator \textit{g} where $ g \in {Z^\*}\_{n^2} $ and $ gcd(\frac{g^\lambda mod {n^2} - 1}{n}, n) = 1 $.

\begin{center}

\textit{g} = 6497955158

\end{center}

With the generator \textit{g} value selected the modular multiplicative inverse can be calculated and used second value in the private key.

$$ \mu = \phi(n)^{-1}\ mod\ n $$

$$ = \Bigg( \frac{6497955158^{31536}\ mod\ 16095743161 – 1}{126869}\Bigg) ^{-1}\ mod\ 126869 $$

$$ = 53022 $$

With the Paillier Cryptography the system would also require the number of expected voters that will participate as well as the number of candidates or parties. This means that the

base, used to prepare the encrypted message, has to be greater than the number of voters. [<><><>] The team has decided to implement the system with a small set of voters and candidates which is the same as the article mentioned previously to allow for easier testing.

\begin{center}

\textit{Number of Voters} = 9

\textit{Number of Candidates} = 5

base = 10

\end{center}

PAILLIER ENCRYPTION

Once the user has been verified and their vote submitted, the system will proceed to format the message and then encrypt the result. Ideally this part of the system would be on the client’s side and can be DRE (Direct Recording Electronic) machine at voting stations allowing users to cast votes through a monitor and keyboard or touch screen input. This will ensure that even if the user’s vote is intercepted the middle man will have a harder time decrypting and encrypting. The encrypted vote will then be stored on a file on the server with other votes cast before it.

SETTING VOTE MESSAGES TO BE ENCRYPTED

$$ m = base^0 + base^1 + base^2 ... + base^{Number\ Of\ Candidates\ -\ 1} $$

PUBLIC KEY ENCRYPTION FORMULA

$$ E(m\_i) = c\_i = g^m\_i\ \* {r\_i}^n\ mod\ n^2 $$

The formulas are used prepare and encrypt the each vote message. The only difference in value is \textit{r} in the second formula listed. The system will randomly generate it where $ r \in {Z^\*}\_n $.

For an example using the values outlined in “Setting Up Paillier Crytopsystem” section, if two voters were to cast the same votes they will still be different encrypted values as long as \textit{r} isn’t the same for both of them.

\begin{tabular}{| l | l | l | l | l | l | l |}

\hline

& Alice & Bob & Charlie & Unencrypted Vote & Random \textit{r} & Encrypted Vote\\ \hline

VoterA & & $\surd$ & & $m= 10^1 = 1$ & 35145 & 13039287935 \\ \hline

VoterB & & $\surd$ & & $m= 10^1 = 1$ & 110006 & 11301345415 \\

\hline

\end{tabular}

PAILLIER DECRYPTION

The Paillier encryption allows the votes to be tallied up without having to decrypt each of them individually. [<><><>] This keeps user confidentiality and prevents tempering of votes.

Tallying Formula

$$ T = \Pi^{n\_v}\_{i=1}\ c\_i \ mod\ n^2 $$

PRIVATE KEY DECRYPTION FORMULA

$$ m = L(c^\lambda\ mod\ n^2) \mu\ mod\ n $$

Using the above formulas it presented the team with the accurate voting results as outlined in the test result section of this paper.

STORAGE

The simulation voting results are encrypted and stored within a text file.

OVERALL

With the current state of the system it runs more closely to the DRE.

However, if given the time it can be converted into a web application.

The system can either become the web server or it can be a CGI script that the administrator can run to calculate the tally.

Unfortunately, with the current expertise within the team the entire proposed system could not be implemented within the time frame.

The features that were not implemented includes

Making the system into a web server

Database connection instead of file storage

A more sigfficiated authentication system

Overall the system itself is very simple.

TESTING

Testing was done to find any possible bugs that will ruin the integrity of the votes.

As a result unit tests was done on the encryption and decryption functions of the system.

Unit Test – Decryption

Input:

\begin{tabular}{| l | l | l | l | l | l |}

\hline

Alice & Bob & Charlie & David & Elliot & Encrypted Vote \\ \hline

& $\surd$ & & & & 13039287935 \\ \hline

& & $\surd$ & & $\surd$ & 848742150 \\ \hline

& & & & & 7185465039 \\ \hline

& & & $\surd$ & & 80933260 \\ \hline

$\surd$ & & & $\surd$ & & 722036441 \\ \hline

& $\surd$ & & $\surd$ & & 350667930 \\ \hline

& & $\surd$ & $\surd$ & & 4980449314 \\ \hline

& $\surd$ & & $\surd$ & & 7412822644 \\ \hline

$\surd$ & & & & & 3033281324 \\ \hline

\hline

\end{tabular}

Output:

\begin{tabular}{| l | l | l | l | l | l |}

\hline

& Alice & Bob & Charlie & David & Elliot \\ \hline

Result & 2 & 3 & 2 & 5 & 1 \\ \hline

Expected & 2 & 3 & 2 & 5 & 1 \\

\hline

\end{tabular}

Finally, full scale integration test was done to ensure that the information created from the encryption is decrypted properly.

The team was not able to create a full test on the simulation system. This includes a timing analysis of decrypting mass amounts of votes you would expect a real system would have to go through. However, it can be assumed that it will require a large amount of server resources and time consuming. This issue can be <><><>medigated<><><> by parallelizing the system.