Thomas Simmons, Ricky Matherly, Jalen Keller, Noah Caldwell, Lee Hong

Dr, Elarde

CSCI 4628

April 26, 2023

Group Project Final Paper

Voter fraud is a serious issue facing the American election systems today. While it is often a catch-all term thrown around by political candidates to further an agenda or trigger a recounting of votes, it is in fact a serious issue. Even more so in autocratic nation-states with pseudo-democratic governments. Current voting systems face multiple vulnerabilities in both their underlying infrastructure, their hardware, and their code.

In this paper we will describe our project, namely, the design, advantages, disadvantages, and future research. For our project we intend to develop a secure voting system. To accomplish this we will develop a Java application that provides a GUI for the user to choose from a list of candidates, namely, candidate 1 and candidate 2. Upon submitting, the user’s vote will be logged in a MySQL database, along with the hash. Our project utilizes key defensive programming concepts to enhance the security of the program as much as possible. ChatGPT was used to develop roughly 15% of this project.

According to the FBI, “On July 13, 2018, a federal grand jury sitting in the District of Columbia returned an indictment against 12 Russian military intelligence officers for their alleged roles in interfering with the 2016 United States (U.S.) elections” (FBI.gov). While existing as a hot topic in many circles, there is in fact some substantial evidence of voter fraud in the 2016 and 2020 presidential elections.

Additionally, Russia runs an operation code named “Project Lakhta”, whose sole purpose is, “to disrupt the democratic process and spread distrust towards candidates for political office and the political system in [the United States] (justice.gov). The aforementioned threat actors are part of a larger international cooperation between U.S. adversaries to influence the election system.

An extension of this larger effort to disrupt U.S. election systems is attempts by U.S. adversaries to interfere with the election systems themselves. Our project attempts to serve as an example of how defensive programming concepts are applicable to the development of robust election systems using more applicable technologies to build a fraud proof election system.

The importance of security is one that can not be ignored when building applications. There are many aspects of an application that is vulnerable to hackers with malicious intent. The surface areas for attacks are the database, the connection between user and the database and brute force attacks on the logins/signups windows. We want to make sure that the database server is secure, use a secure connection, and in the future, implement a maximum number of fail attempts on passwords. All of which were used to enhance the security of the overall application.

Whenever an application uses user input, it opens itself up to security vulnerabilities. Hackers will attempt to use surfaces of user input to their advantage. Without user validation, hackers can inject malicious code into the application. Input validation is also important for non-malicious users. Common users may not give the input developers are expecting. This can lead to unintended consequences and potentially break the program. Good input validation should account for every possible scenario, especially when it could lead to a security breach. Input validation should be more important than the user experience.

The purpose of this project is not to prove that we can make a voting system, but that we can make a secure voting system. This means input validation plays an important role in ensuring the security of our application. We will use input validation to prevent users from manipulating our voting system. User experience will not be sacrificed in exchange for any potential security threat. We will be checking that the input is of the required type, of the expected length, and of the necessary authentication. If users are allowed to submit multiple votes or injected code, the security of the election would be compromised. Key techniques we will be implementing include sanitization, buffer overflow prevention, and relevant error messaging.

As mentioned above, our project attempts only to serve as a demonstration of defensive programming principles. The application consists of a Swing GUI, server side code to connect to a MySQL database, and the MySQL database itself. The program consists of four classes, namely, VotingApp (main class), createAccount, Login, and Ballot. The classes function as follows:

* VotingApp:
  + The VotingApp class instantiates the program. When instantiated the VotingApp class creates a new JFrame consisting of two buttons: Login, and Sign Up. The Login button creates a new instance of the Login Class and then calls the showLogin method. The showLogin method displays the Login screen. The Sign Up button creates a new instance of the createAccount class and calls the showCreateAccount method.
* createAccount:
  + Once the showCreateAccount method is called, a new JFrame displays with two fields: Username and Password. In this GUI element, the user is to enter a desired username and password. When the desired username and password is entered, the login information will be logged to the database, a “Account creation successful” message is displayed, and the user is redirected to the login screen via the showLogin method.
  + User input for the createAccount class is validated against a regular expression to ensure 1) an eight character alphanumeric password is entered, and 2) No SQL statements are allowed to be submitted to the database. The username field is also checked against input validation to ensure that a username of at least 1 alphanumeric character and no more than 20 alphanumeric characters are entered. This serves the dual purpose of ensuring that no SQL injection is possible by eliminating the option for special characters (i.e, semicolon or space) required to execute SQL statements.
  + By limiting the number of characters allowed as input, we also mitigate the risk of a buffer overflow attack on the system.
* Login
  + The Login class is a mirror image of the createAccount class with a few minor modifications. The Login class uses the same input validation as the createAccount class, however reads from the database instead of writing to it.
* Ballot
  + The Ballot class allows the user to select a candidate using a radio button. This prevents the user from being able to select two candidates. The candidate, the username, and the vote hash are then logged to the database.

Java GUI is a crucial component when creating the presidential voting system, and the central part is to display the entire election. For this, we use the Java Swing Framework to accomplish this.

JSwing is a toolkit in Java used for creating graphical user interfaces (GUIs) for desktop applications. It provides a set of components such as buttons, text fields, and menus that can be customized in appearance. Swing also supports layout managers and events and listeners for user interaction. It is known for its ability to provide a consistent look and feel across multiple platforms.

The GUI for this project can be summarized into four major parts. Panels, Frames, Labels, and Buttons:

1. Panels are the main background for GUI, and it is displayed with a white background, making it clearer to see. Different sizes, ranging from the X and Y axis, can change the Panel's position. The vital point for the Panel in the voting system is to display the whole layout for the entire election. However, you can change the color with setBackground with a color variable for changing the background.
2. Frames are similar to Panels; however, it is used for Labels and Buttons to be displayed upon them. Frames can be an alternative to Panels, combined to show a complete GUI. The primary point of Frames is to say the election while utilizing buttons and labels.
3. Labels are used for background text format. For this instance, it is to display the names, the title, and other details.
4. Buttons are used to initiate a function, most occasionally an action listener. The action listener, on most occasions, displays a message telling the user that the process has worked.

Databases are the biggest part of storing all kinds of information. It’s used by almost every company in the world to store things like company information or even customer information. Databases have to be secured to the best of your ability because all information can be used by somebody and they will take advantage if it’s possible to. There will be so much potential money being lost if you don’t make the smart investment of paying for a secure database.

The database will need to be secure to prevent unauthorized reads or writes. The database will be where we store sensitive personal information as well as the tally of votes for each candidate. The candidate a user votes for will stay anonymous in accordance with the law. We will not keep track of this information because of any potential data breach. The database will be hosted using XAMPP for this project along with an Apache web server on our local machines. All of the information will be secured tightly within a MySQL database.

The connection to the database will need to be protected to prevent any sniffing, man-in-the-middle attacks, or any other malicious intrusions. We will focus on heavy backend security to ensure the security of our database queries. The information will be separated in different columns like: Candidate names and votes. The database should be very accessible to the rest of the code.

We can explore the many ways to ensure that each user/voter is only able to vote once. We can look at options like geo-location and IP-address tracking but in this day and age, having multiple computers is not unheard of and almost all devices are portable now. We can enhance the authentication for users to ensure that each person can have only one account. Gathering information like Social Security Numbers and other personal information can assist with this. Though this solution is not full proof, we can compound other authentication methods to perform a more secure authentication process for our users. Other methods include but are not limited to:

User authentication: Users can be required to authenticate themselves using their unique login credentials such as username and password, or biometric authentication. This will ensure that each user is only able to vote once.

1. Unique identification codes: Each user can be assigned a unique identification code that can be used to validate their vote. This code can be sent to the user's email address or mobile phone and can be used to verify their identity when they cast their vote.
2. IP address tracking: The online voting system can track the IP addresses of users who cast their vote. This can help to identify users who attempt to cast multiple votes using different accounts or devices.
3. Time-stamping: Each vote can be time-stamped, so that if a user tries to cast a second vote, the system will recognize that the vote has already been cast and will not allow the user to cast a second vote.

Two-step authentication can improve the security of an online voting system. Two-step authentication requires the user to provide two forms of authentication before accessing the system. This can include a password or PIN as well as a secondary factor, such as a fingerprint or a one-time code sent to their mobile phone.

Implementing two-step authentication in an online voting system can make it more difficult for unauthorized users to access the system or cast votes on behalf of other users. It can also help to prevent bots or automated scripts from casting fraudulent votes.

However, it's important to note that two-step authentication is not foolproof and can still be vulnerable to attacks such as phishing or social engineering. Therefore, it should be used in conjunction with other security measures, such as encryption and regular security audits, to ensure the integrity of the voting system.

Blockchain can help with voting by providing a secure and transparent way to record and store votes. It eliminates the need for intermediaries, increases accessibility, and boosts voter confidence in the electoral process. However, implementing a blockchain-based voting system would require careful planning and coordination to ensure its effectiveness and security.

1. Blockchain records votes securely and transparently, making it difficult to manipulate results.
2. All votes in a blockchain-based voting system can be seen by anyone, increasing transparency and accountability.
3. A blockchain-based voting system eliminates the need for intermediaries, reducing the possibility of fraud and ensuring accurate results.
4. Blockchain-based voting systems are accessible from anywhere with an internet connection, benefiting those with disabilities or living in remote areas.
5. By providing a secure and transparent system, blockchain can increase voter confidence, leading to higher turnout and greater participation in democracy.

Works Cited

Seacord, Robert C. Secure Coding in C and C++. 2nd ed., Addison-Wesley Professional, 2013.

OWASP Foundation. "OWASP Secure Coding Practices Quick Reference Guide." OWASP,

2021, <https://owasp.org/Top10/A7_2021-Cross-Site_Scripting_(XSS).html#references>

Java Documentation. <https://docs.oracle.com/en/java/>

Java Manual. <https://www.java.com/en/download/manual.jsp>

The GitHub for our project can be found at: <https://github.com/Tsimmsz/SecureVotingJava>