Secure Banking System

CSE 545: Software Security

Amit Kumar
Computer Science (MCS),
Ira A. Fulton School of Engineering,
Arizona State University,
Tempe, AZ, United States,
akuma512@asu.edu

***Abstract*— This document is final project report for the course project Secure Banking System for course CSE 545 Software Security by Dr. Stephen S.Yau. The purpose of the project was to develop an online banking system with the main concentration on security. The report highlights complete details of functionality, system design and key implementations that were used in the development of the secure banking system. It also covers the common vulnerabilities, attack patterns and the prevention techniques used to prevent them.**

I. PROJECT INTRODUCTION

The course had demanded online secure banking system, here onward referred as Cardinals Banking System, which is a software system developed to facilitate secure banking transactions and user account management via the internet. The Cardinals Banking System facilitates the banking of any common bank, which includes all the basic functionalities like viewing, transactions, downloadable banking statements, account details, money transfers, and credit card transactions. The banking organization can use it to track various operations performed by different users of the bank. The system provides a web-based user interface that would allow access to the system at any place and time, with the availability of internet access and a compatible web browser. After verifying the login credentials of the user, the user is taken to a home page where s/he has the complete list of transactions which s/he can perform with the bank. It is broadly demonstrated for two types of users, Internal Users, and External Users. Internal users are bank employees and external users are customers.

As, the system deals with people's personal information (PII) as well as their funds, security is very important in this case. Different vulnerabilities and malicious attacks have been prevented by incorporating different techniques into each phase of software development lifecycle. A complete and comprehensive all round security has been implemented in the system. Some of the implemented security features are SSL, PKI, virtual keyboard, captcha, and OTP. All the required user communication is done through e-mail. The cardinal banking system includes all the security and functionality features mentioned in the requirement document. It strives to protect all the user data from malicious attackers.

The system is made highly secure by implementing different security features. The system is successfully deployed over the network and is functional.

II. SYSTEM DESIGN

Cardinals Banking System is designed to allow internal and external users of the banking organization to perform bank operations online in a secure manner. Security to such application is utmost importance. The Cardinals Banking System was successfully implemented leveraging the Spring MVC framework for frontend as well as backend. The system uses JavaScript, HTML, and CSS for the front-end. The backend was developed using Java and MySQL as a database. We used SQL workbench to easily track the changes being made to the user databases. The controller is used for routing user requests. The View- Front End renders responses back to the user. The Model defines the schema of the database. We implemented the service-oriented architecture in a blend with spring MVC. The architecture includes the service layer that specifies business logic and interacts with Data Access Object for operations on the database. Schema of the database is defined in another layer of models.

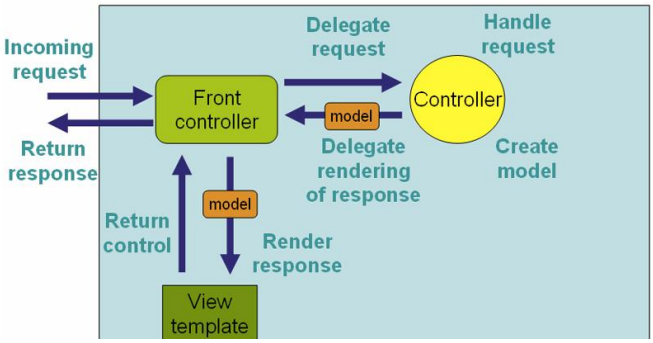

Figure 1 MVC Architecture

We have always kept security in the centre of consideration to make any decisions such as design, tools to used, testing etc. We made sure that at each stage of software development we maintained security standards and incorporated various prevention methods and techniques. SSL and PKI are implemented in order to make secure communications between clients and server. All passwords are encrypted so that CSRF attacks can be avoided. Virtual Keyboard is implemented to prevent user from key logger attacks. Device trust management is done to notify user wrong access of his/her account. SQL injection, Denial of service attacks, Cross site scripting are prevented by implementing three layer input validation and sanitization.

Security testing is done throughout the software development lifecycle. This improves our application performance and robustness from vulnerabilities.

III. SECURITY FEATURE IMPLEMENTATION

Secure banking environment aims at accomplishing a secure and effective platform for managing accounts for normal users and client over the internet for various purposes revolving around their accounts. The Cardinals Banking System is secure and reliable banking system which can withstand and function in an appropriate manner even in the time of an immediate threat or attack. The software satisfied all the necessary functional requirements and security. Overall the software functioned perfectly and there were no major issues in the software, because of implementation of following security features:

Public Key Infrastructure (PKI) supports the distribution and identification of public encryption keys, enabling users and computers to both secure exchange data over networks such as the Internet and verify the identity of the other party.

OTP (One-Time password) is generated for the critical transaction. The generated OTP is valid only once.

Virtual Keyboard prevents the users from keylogger attack. The virtual keyboard is implemented for critical transactions.

Captcha – We are using google reCAPTCHA to rejects automated scripts. The validation is done on server side.

Password Recovery is a mechanism is used to recover the password in case if a user forgets his/her password. If a user requests for forget password mechanism, the system sends a recovery link to registered email.

SSL Connection - Sensitive information such as user credentials, account number, card details of the user are exchanged by encrypting and decrypting the data between bank and user.

HTTPS is a protocol for secure correspondence over a computer network which is generally utilized on the internet. The HTTPS enables transmission of data over any secured layers.

Input Validation – All the given user input are validated. Three layers of input validation are done in the system. One is at the front end in order to validate all the inputs given by the user, another one as middleware (service layer) and the last one is at the back end in order to restrict malicious injection of input into the database. This is to prevent attacks like Cross-Site scripting and SQL injection attacks.

User Authentication - The system validates the user credentials by checking it with the user databases and authenticates the user. Only registered users pass this validation. Unauthorized users are restricted from the system using this mechanism. The database is encrypted.

Personally identifiable information (PII) access of PII information is restrictive. Only user with proper privilege (administrators) can view the view PII of user.

Session Management – HTTP session is used to verify the user and keep track of the session. If the user forgets to log out the open account gets logged out after a certain time. This prevents an attacker from taking advantage of an open account and also helps the user keep their account safe if they forgot to log out.

Invalid authentication – The account gets locked after three unsuccessful login attempts.

Device trust management – If the user logins from a new device a notification is sent to his/her email. The user will be notified in case s/he has not login.

IV. USERS AND SYSTEM FUNCTIONALITIES

A user should be able to use this system from any place and at any time with the availability of internet access and web browser. Users are categorized into 2 types, Internal Users, and External Users. Internal Users will include regular employees, system managers, and account administrators, while External Users will be comprised of individual customers and merchants and/or organizations.

A regular employee is responsible for low priority banking operations such as authorizing non-critical transactions, create, modify and delete external users account upon having authorization.

A system manager is responsible for higher priority banking operations and responsible for the authorization of critical transactional operations. S/he can manage external users can authorize critical/non-critical transactions etc.

Administrators maintain all the internal user accounts and the banking system. S/he can access PII, access system logs, manage internal users etc.

Individual customers are those who have a user account for performing personal banking transactions such as personal fund transfer, debit, and credit from the personal user account.

Merchants/Organizations are users having specialized banking transaction processing requirements, such as client payment processing.

User Account Management - Internal users can manage all externals user accounts i.e., view/modify their personal and

account details and credit/debit money from their account based on access. Only admin can view the personally identifiable information of external users.

Savings and Checking account- A user can have saving account or credit account or both.

Fund Transfer - The external user can transfer fund to his/her accounts, to another user account upon authorization from internal users. Fund transfer can be done by giving valid email of the customer. The merchant can make payment on behalf of users.

Credit card- The customer can use the credit card and needs to pay the due at the end of every month. The customer can check his/her credit card details by logging into the system.

Banking statements - An external user can download banking statements as pdf. S/he can download the statement of accounts as well as credit card.

V. DESCRIPTION OF MY CONTRIBUTION TO THE PROJECT

I was the lead of the security team and hence was responsible for the design decisions of the project and major contribution in securing the application. I have been actively involved in all stages of the project which includes requirement specification, documentation, deciding the technology and architecture, work allocation, testing and fixing bugs. I lead the group of 4 members to implement the required security features. I have done extensively researched and explored the security features to be implemented. I prepared test cases and class diagram. I developed the encryption for the PKI. I have implemented a service which was generating the key pair i.e., private and public key at the time of user registration. The public key is enclosed in an email and sent to the user, which client uses to authenticate the server. In order to make sure that server is talking to correct client, the server uses the private key to authenticate the client. I also implemented the PDF generator using the iText library to facilitate user the feature to download and view their account information. I also implemented the virtual numpad as a security feature using javascript, which allows the user to enter OTP during the critical transaction. This feature saves the user from key logger vulnerability. Being sub-leader I always keep the group leader informed of task accomplishment, issues and status. I helped a teammate in the implementation of device trust management. I also performed unit testing, integration testing, system testing of the project and found the vulnerabilities and then subsequently fixed the bugs.

V. LESSONS LEARNT

I learned how to design secure web applications, understanding the software requirements along with security constraints is the key. I have learned what is PKI, SSL, HTTPS, OTP etc. I learned we should not trust user input and validate and sanitize all user input this may lead to vulnerabilities such as SQL injection, Cross Site Scripting etc. I learned on how to use Git through this project. I also learned a new scripting language 'JavaScript'. In addition to that, I also learned a lot about how to defend the application of common attack patterns such as username and password sniffing, Cross site scripting attacks, SQL injection and also CSRF attacks. The importance of proper session management and error handling i.e. stack trace shouldn't be visible to the user and the software should fail securely. All data must be encrypted to avoid loss of critical information. I learned why to incorporate security early in the software development life cycle and importance of security testing throughout the software development lifecycle.

ACKNOWLEDGEMENT

I would like to thank Professor SS Yau and the Teaching Assistant Yaozhong Song for their immense support during the entire course of the project and the coursework. I would also like to thank my all team members for giving their best in completing this project successfully.

TEAM MEMBERS

Manasa Guntaka, Ayush Gupta, Madhulahari Illuri, Shivani Jhunjhunwala, Aastha Khanna, Jainesh Kothari, Nikhil Loney, Joel Mascarenhas, Sagar Navgire, Akhila Polisetty, Nishtha Punjabi, Sushant Sakolkar, Mitikaa Sama, Kevin Bharat Vira

REFERENCES

[1] Spring MVC framework, [online] Source: https://spring.io/guides/gs/serving-web-content/

[2] iText Pdf, [online] http://developers.itextpdf.com/downloads

[3] Cross-Site request forgery, [online] https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

[4] Public Key Infrastructure, [online] https://en.wikipedia.org/wiki/Public_key_infrastructure

[5] CSE545, Lecture slides by Dr. Stephen S.Yau, fall 2016, Arizona State University, Tempe.