Secure Banking Environment Using AWS Cloud and TA-Cipher Algorithm

[1]Anshuman A. Kumar, [2]Tanaaz Mansoori

Department of Information Technology

Dhole Patil College of Engineering, Near EON IT Park,
Vitthal Nagar, Kharadi, Pune, Maharashtra 412207.

Abstract : In this paper , the key idea is to provide online banking services to the user which is secure. This will be accessible to all customers who have a valid User Id and Password. This is an approach to provide an opportunity to the customers to have some important transactions to be done from where they are at present without moving to bank. We are going to deal the existing facts in the bank i.e.; the transactions which takes place between customer and bank. We provide a real time environment for the existing system in the bank. We deal in the method transaction in the bank can be made faster and easier that is our project is an internet based computerized approach towards banking. As the application of project is regarding internet banking ,we used JAVA a simple, object-oriented, network savvy , interpreted, robust, secure, architecture neutral, portable, high performance, multithreaded dynamic language. Amazon Web Services RDS managed relational database service is used to store all the data using MYSQL.

KEYWORDS: Secure data transmission, Encryption, AWS RDS, MD5, Java, Data Security, Online Banking.

I. Introduction

The Traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transactions he has to go to bank and perform the necessary actions, which may not be so feasible all the time. It may be a hard-hitting task for the users and the bankers too. The project gives real life understanding of Online Banking System and activities performed by various roles in the supply chain.

Nowadays, banking system has been more accessible than ever to perform essential duties e.g online transfer, change account information without having the need to visit nearby bank and perform all of them through online. While the online access to banking functionality for users has given great flexibility but it requires sufficient security provided by the banking company to help protect from unauthorized access to user's accounts and also help the user to perform his activity securely maintaining the privacy.

Objective -

- To design such banking application securely according to design guidelines set in the course.
- To perform testing for verifying functionalities.
- To provide online banking customers the facilities to access and manage their bank accounts easily and globally.

Basic Concept -

The SECURE BANKING SYSTEM is a web application, compatible with any web browser with access to Internet, providing online banking services.

- Services are provided to the customers of the bank.
- Ensures security, integrity and confidentiality of customer's activity and personal information from unauthorized access.
- A two-factor authentication is setup for individual users to sign into the application.
- First factor is by supplying username, password and second factor is using One Time Password (OTP).
- Amazon RDS is used to store MYSQL database which makes it easy to use to enhance availability and reliability for production workloads.

II. LITERATURE SURVEY

[1] Gotimukul Venkatesh , Sunkara Venu Gopal , Mrudula Meduri , Sindu C, “Appication Of Session Login And One Time Password In Fund Transfer System Using RSA Algorithm” – 2017

Password thefts, a serious security threat to Internet users where the perpetrator steals the secure password and misuses it which makes the individual look like legitimate user, in an order to gather personal and financial information. It is important to prevent such identity theft attacks especially in fund transfer. One of the ways to prevent the password theft is to authenticate the user. Creating a high secure password can achieve high level of security in authenticating the user over the internet. Using the instant mailing service available in internet, user will obtain the public key. Then the private key is shared using secure server client connection ensuring legitimate user receives the private key. Both the keys are generated as RSA Tokens using Random password number Generator which are synchronized between server and client and the login expires once the current time limit exceeds the access time. The main aim is to use session login narrowing the login time thereby reducing the probability of hacking. These algorithms are very economical to implement provided they are time synchronized with the user.

[2] Priyanka Vora , Kranti Sonawane , Sneha Phulpagar , Prof.A.P.Ydav, "Data Security Using Colours and Armstrong Numbers" - 2016
In real world, data security plays an vital role where security, privacy, validation, integrity, non-repudiation is given importance. There are some common techniques used for secure data transmission over network.This paper provides a technique for data security which encrypt the data using a key involving Armstrong numbers and colours as the password. Three set of keys is used to provide secure data transmission with the colours acting as vital security element thereby providing authentication.

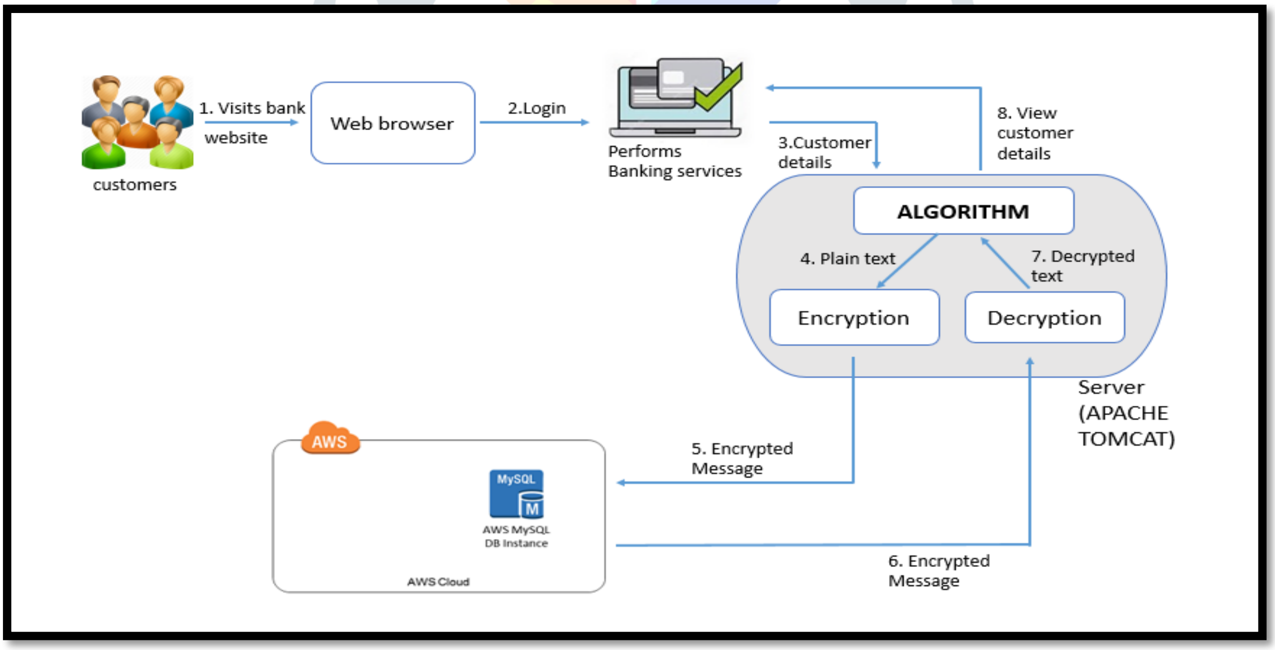
[3] Jhumkee Iyengar , Manisha Belvalkar "Case Study of Online Banking in India:User Behaviors and Design Guidelines "
This paper documents online banking trends, behaviors and expectations of Indian consumers and banks. It is based on excerpts of a large industry case study of users from 4 leading banks. While banks view online banking essentially as a technology solution, it is a relatively new area for Indian consumers and not yet self-supporting. Being a savings based culture still, Indian consumers are cautious about their financial assets. They are also relatively recent entrants to internet based services. Design of these systems must therefore be based on an understanding of these users' outlook and priorities through task centric, security assured and service oriented solutions minus the technological challenges. Design lessons suggest viewing online banking not just as a convenience alone anymore but beyond it, to provide service, simplicity and security. This will create satisfied online banking customers and therefore profitability for the bank.

[4] Ramesh K , Ramesh S "Implementing One Time Password Based Security Mechanism for Securing Personal Health Records in Cloud " – 2014
Personal Health Record (PHR) is an emerging patient-centric model of health information exchange, which is often outsourced to be stored at a third party, such as cloud providers. Researches had been taken up in enhancing the security of the out sourced PHRs by implementing the Attribute Based Encryption (ABE) to assure the patients control over their PHRs. Yet, most of the attribute based encryption introduces complexity in access control, key management, privacy exposure and scalability of the System. However, the authentication module of the PHR system had been researched less when compared to the security module of the system. In this paper, we propose a "One Time Password" user authentication module along with Advanced Encryption Standard (AES) for encrypting the owners personal Health Record before uploading it onto the semi trusted cloud server.

III. PROBLEM STATEMENT

To develop a novel application for secure banking system using our own developed encryption algorithm- TA cipher, Java, MYSQL database stored in AWS cloud using Amazon RDS in order to make the online banking services secure , feasible and user friendly.

IV. SYSTEM ARCHITECTURE



The Secure Online Banking System is a web application, compatible with any web browser with access to Internet, providing online banking services for the customers of the bank. A two-factor authentication is setup for individual users to sign into the application :-

1) First Factor :- supplying username, password
2) Second Factor :- using One Time Password (OTP)

While signing in, an OTP will be generated and sent to the users email to verify his authenticity. Generated OTP is a time based and is valid for only 2 minutes. Passwords will not be stored in the database as plain text. Instead they will be passed through standard MD5 hash function and digest is stored in the database. After 3 unsuccessful attempts of logging into account, customers account will be blocked. User is automatically logged out when inactive for more than 3 minutes in the website and session is invalidated. Algorithm developed by us is used to encrypt and decrypt the sensitive information of the customers and the bank sent through the server. Unique customer id is generated and provided to the customers after successful online account creation.

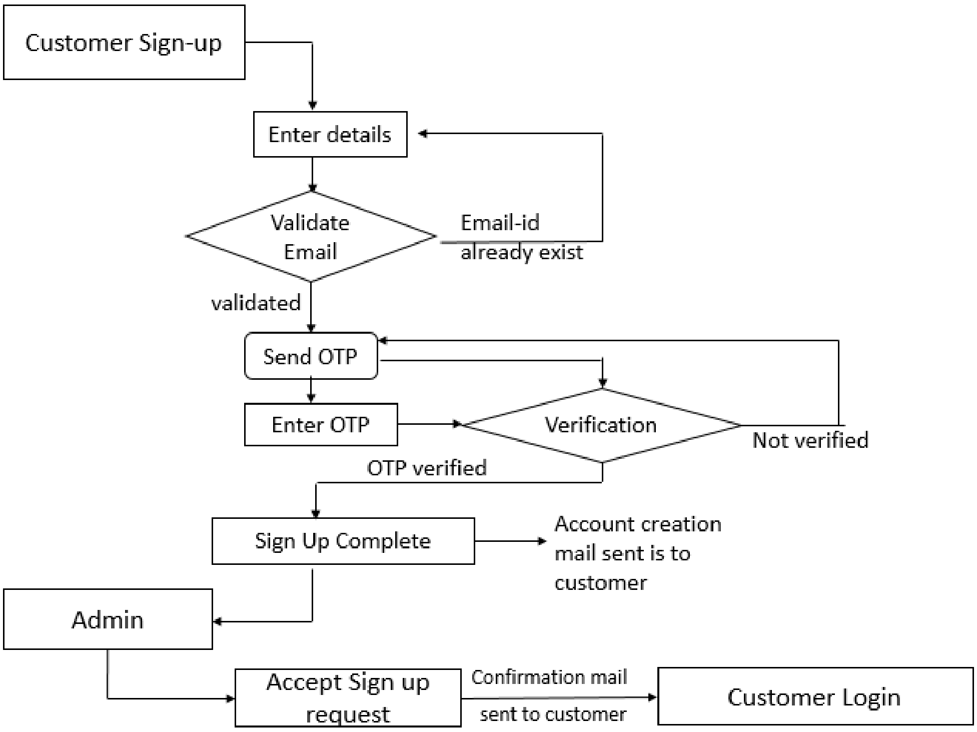
It consists of 2 modules: Admin Module and Customer Module.
Admin Module:
It consist of-

- Accept customer's online account request.
- View customers
- Edit customer details
- Block/Unblock account
- View all customer's transaction
- Debit , Credit and Funds Transfer

Customer Module:
It consist of-

- View account details
- Funds transfer
- View statement
- Mini statement
- Check balance

V. FLOW DIAGRAM



AWS RDS -
Amazon RDS increases the ease of access and availability of Database in our Project while ensuring to provide top of the line security.

- The automated backup feature of Amazon RDS enables point-in-time recovery for our database instance. Amazon RDS will backup our database and transaction logs and store both for a user-specified retention period.
- Amazon RDS will automatically replace the compute instance powering our deployment in the event of a hardware failure.
- Amazon RDS allows us to encrypt our databases using keys we manage through AWS Key Management Service (KMS). On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots.
- Amazon RDS supports the use of SSL to secure data in transit.
- We can scale the compute and memory resources powering your deployment up or down, up to a maximum of 32 vCPUs and 244 GiB of RAM.

MD5 -
The MD5 function is a cryptographic algorithm that takes an input of arbitrary length and produces a message digest that is 128 bits long. The digest is sometimes also called the "hash" or "fingerprint" of the input. MD5 is used in many situations where a potentially long message needs to be processed and/or compared quickly. The algorithm takes as input a message of arbitrary length and produces as output a 128-bit 'fingerprint' or 'message digest' of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given pre-specified target message digest. The MD5 algorithm is intended for digital signature applications, where a large file must be 'compressed' in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA.

VI. ALGORITHM

Following Are Steps for the Encryption module-

STEP - 1 Conversion of Plaintext

i. Calculate the no. of digits i.e. length of plaintext.
ii. Convert Plaintext to ASCII number.

STEP - 2 Key Generation

Generate a random Number (key) Such that if ,
Number of digits in key = n
Then, Number of characters in plaintext should be <= n*n

STEP - 3 Key Modification and Addition

i. Generate the selected key to the length of n*n while incrementing the power of digits after every round, namely var k .
ii. Add the plaintext ASCII number with the generated key, namely var p.

STEP - 4 Matrix Generation

i. Create a matrix of var p using dimensions [n*n] where n is the length of the key selected, namely R matrix.
ii. Convert var k in the form of matrix of dimensions [n*n] while incrementing the power of digits in each row, namely K matrix.

STEP - 5 Matrix Addition and Transposing

i. Add R matrix with K matrix and produce the resultant R1 matrix.
R[] + K[] = R1[]
ii. Add R1 matrix with K transpose matrix (K') and produce resultant new R matrix.
R1[] + K'[] = *new* R[]
iii. Continue sub points i) and ii) till the completion of no. of rounds.
new R[] + K[] = *new* R1[]
…..
…….
iv. No. of rounds is determined by the length of key.
No. of rounds = length of key(n)
v. Final matrix is M matrix.

STEP - 6 Cipher text

i. Transpose M matrix , namely M' matrix.
ii. Send M' matrix [Encrypted value to the receiver in the form of message].

Following Are Steps for the decryption module-

STEP - 1 Cipher text and Key Matrix Generation

i. Calculate the no. of digits i.e. length of key as n.
ii. Convert cipher text in the form of matrix of dimensions [n*n], namely M' matrix.
iii. Convert key in the form of matrix of dimensions [n*n] while incrementing the power of digits in each row, namely K matrix.

STEP - 2 Matrix Subtraction and Transposing

i. Transpose M' matrix , namely M matrix.
ii. Transpose K matrix , namely K' matrix.
iii. If key length i.e. n = odd , then
Subtract M matrix with K matrix and produce resultant M1 matrix
M[] – K[] = M1[]
Else
Subtract M matrix with K' matrix and produce resultant M1 matrix
M[] – K'[] = M1[]
iv. Continue subtracting resultant matrix with key matrix in each round while transposing K matrix in alternate round.
v. No. of rounds is determined by the length of key.

No. of rounds = length of key(n)

vi. Final matrix is R matrix.

STEP - 3 Encrypted and Key Matrix Conversion

i. Convert the matrix R in the form of variable , namely var c.

ii. Convert the matrix K in the form of variable of length of n*n while incrementing the power of digits after every round, namely var k.

STEP - 4 Subtraction

i. Subtract var c with var k producing , namely var p.

STEP - 5 Plaintext

i. Convert var p to string, decrypting it back to plaintext.

VII. RESULT AND CONCLUSION

In this study, we have implemented a Java based online banking system using a self-developed encryption algorithm TA cipher. Data incoming from the website is being encrypted on the server side using TA Cipher before being updated in the database. We have used Amazon RDS as a cloud database which provides scalability, improved availability and security in data handling. The Data transmission between the server and database is secured by SSL encryption provided by Amazon Web Services. We use a two factor authentication for customer authentication and verification. Admin has the right to approve to disapprove any customer after personal authentication. After login the customer can use various banking services provided. Admin acts as an over watch and has access to all the details of each customer. We aim at providing a more secure banking environment using Cloud services and TA Cipher and other security mechanism in our prototype.

VIII. FUTURE ENHANCEMENTS AND SCOPE

As we have made a prototype in the limited time we had we suggest some future enhancements for a better and more customer friendly banking environment. Firstly, the inclusion of SMS OTP and a responsive website for mobile phone users. Second, the use of Payment gateway API's like PayPal for Bill payments. Third, Use of SSL certificate for the website to make it more secure (using https website).Fourth, Use of Virtual Keyboards. And Fifth, Use of Biometrics for security(Face and Iris recognition).

REFERENCE

[1] Gotimukul Venkatesh , Sunkara Venu Gopal , Mrudula Meduri , Sindu C "Appication Of Session Login And One Time Password In Fund Transfer System Using RSA Algorithm" - 2017

[2] Priyanka Vora , Kranti Sonawane , Sneha Phulpagar , Prof.A.P.Ydav "Data Security Using Colours and Armstrong Numbers" - 2016

[3] Jhumkee Iyengar , Manisha Belvalkar " Case Study of Online Banking in India: User Behaviors and Design Guidelines"

[4] Ramesh K , Ramesh S "Implementing One Time Password Based Security Mechanism for Securing Personal Health Records in Cloud " – 2014

[5] Hyun-Chul Kim1, Hong-Woo Lee1, Kyung-Seok Lee 1, Moon-Seog Jun1 "A Design of One-Time Password Mechanism using Public Key Infrastructure"

[6] Luigi Maria Bottasso "A Public-Key Cryptography Tool for Personal Use A Real- world Implementation of ECC for Secure File Exchange"

[7] Priyanka Patel1, Mitixa Parmar2 "Improve Heuristics for User Session Identification through Web Server Log in Web Usage Mining