Password Storage and Retrieval

Nitin K N[1] Sharath S Chellappa[2] Chetan Purohit[3]
Madhusudan Thyagarajan[4]

[1]USN-1PI13EC064 Electronics and Communication Department
PES Institute of Technology

[2]USN-1PI13EC092 Electronics and Communication Department
PES Institute of Technology

[3]USN-1PI13EC025 Electronics and Communication Department
PES Institute of Technology

[4]Professor Electronics and Communication Department
PES Institute of Technology

Feasibility Report, 2017

Outline

Introduction

- LinkedIn hack in 2012 was a shocking event to the entire world. The hack involved passwords of nearly 6.5 million users being stolen by Russian cybercriminals. The hash of the passwords was stored on the servers which was tackled using the Rainbow table attack.
- Adobe too had a data breach in the product of Lastpass which is its own password manager.
- Companies like Yahoo(2013 and 2014), Sony Online Entertainment(2011) and Epsilon too have had breaches of login details and passwords multiple times over the years.

Problem Statement

To design a bank system which ensures that compromise of any one server storing the password, never gives away the entire password of the user.

Objective

- The objective is to develop an app for a banking system where the user enters his password which is never completely stored anywhere.
- The Bank server splits the hashed password given to it into different pieces and stores these in geographically different locations.
- Nowhere does the Bank server or the Servers storing the pieces of password get to know what the complete password is. Hence compromise of the servers would never give away the users password as it is.
- The adversary would require multiple points of attacks in order to break the password.

Splitting the password

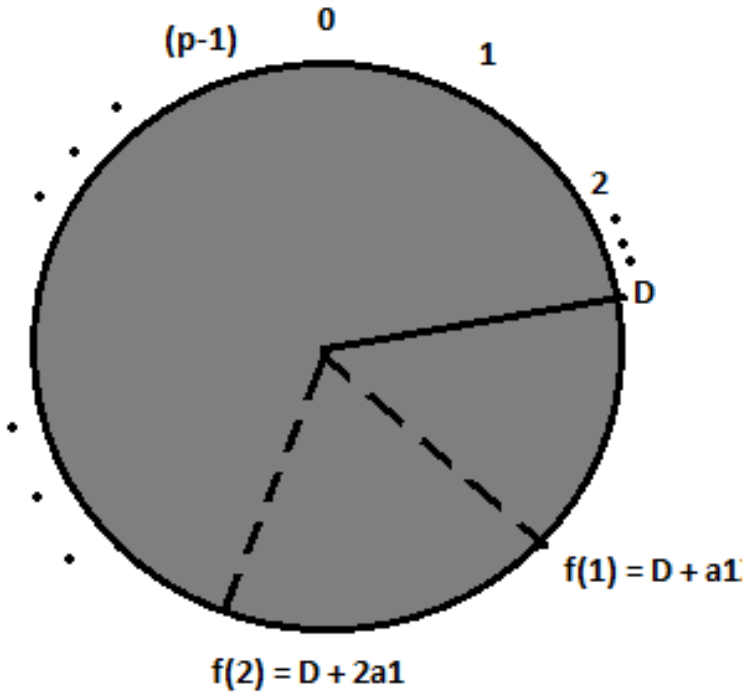
How to share a secret, A. Shamir

- Its a (k, n) threshold scheme where the password D is split into n pieces out of which atleast k pieces are required to reconstruct the secret D. Let the password be $D < p$ where p is a prime number.
- Choose $(k-1)$ random positive integers $a_0, a_1, a_2, \ldots a_{(k-1)}$ with $a_i < p$.
- Let $a0 = D$. Build the polynomial $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{(k-1)} x^{(k-1)}$. Choose $n < p$ points out of this polynomial by putting $x = 1, 2, 3 \ldots n$. The n ordered pairs $(x,\ f(x)\ mod\ p)$ are the n pieces of passwords.
- Any k of them can be used to solve the polynomial and find $a_0 = D$ which is the password. (y -intercept)

Splitting the password

Application in our project

- Our project uses this algorithm in the specific case of $(2, 2)$ scheme.
- Password is split into 2 pieces. These are 2 points on a line. Both are required to obtain the password D which is the y-intercept.



Literature Survey

The recent attacks on the servers of tech giants, brings to light the fact that newer and better mechanisms for storage are in dire need. The rainbow table attack, carried out by the hackers, showed that a widely-used mechanism such as hashing[6] is not perfect and needs optimization. On researching for alternate mechanisms, a paper by Adi Shamir, titled "how to share a secret"[2], highlighted a mechanism of password splitting. Extending this idea to a scenario of a bank and its users, we propose to break down a user given password and store these pieces in multiple servers. This paper is what the entire project is based on.
Further security over the communication channel has been taken care of by RSA algorithm[3] used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm which means that there are two different keys, public and private. The algorithm is based on the basic statistical problem that finding the factors of an integer is hard (the factoring problem)[4].

Architecture

Pre-requisites

- The app is device specific. The user must register his device with the Bank before hand in order to avail the service.
- Upon Registration, the user will be given the app with the unique login id hardcoded in it.
- This login id is unknown to the user and is valid only for the particular device.
- Misplacing the device would mean that the user would have to revisit the bank and repeat the procedure for registering another device.

Architecture

Initial Password Splitting

User is asked to enter his registered bank account number and name.

$E_{skb}\{E_{pku}\{\text{Login id} + \text{bank account number} + \text{name}\}\}$

OTP + Security Question

OTP + Security Question

The Bank Server checks the answer and now splits the password and stores the different pieces in different servers with the hashed Login id being the query point. Location pass is generated conveys location of servers.

OTP + Answer + Password

$E_{skb}\{E_{pku}\{\text{OTP} + \text{User Password} + \text{login id}\}\}$

User Password Piece + Hashed(login id)

The user gets an acknowledgement that his password has been accepted.

Location pass to be hard coded into the app

Architecture

Password Retrieval

User clicks on the button to request access to his account

$E_{skb}\{E_{pku}\{\text{Login id}\}\}$

OTP

OTP

The Bank server correlates OTP with the login id. This correlation expires in a finite time.

The Bank Server decrypts the packet to get the login id and OTP, gets the location of the servers having pieces from the Location Pass and sends the login Id to each of the servers after checking if the OTP matches.

OTP + Password

$E_{skb}\{E_{pku}\{\text{OTP+login ID + Hash(User Password+OTP) + location pass}\}\}$

Hash(Login id) to each server

The user gets to access his account over a secure connection.

Returns Port number to set up connection with if there is a match

Each piece is sent to Bank Server

Protection against compromise of any stakeholder

Compromise of mobile application

- Using a keylogger would undoubtedly allow the hacker to get the password of the user.
- This password can only be of use if the adversary has the device because the location pass and login id is hardcoded into the device.
- This would mean that the adversary would have to obtain the device from the user along with using the keylogger to cause damage to the users data.

Protection against compromise of any stakeholder

Compromise of bank server

- By compromising the bank server, the adversary gets to know the following,
  - Location pass
  - OTP
  - Hashed User Password + OTP
  - Login id

  Location pass OTP Hashed User Password + OTP Login id
- With the given login id, he would have to dereference the registered device and reference his own device so that the connection is established with his server.
- Assuming the user password + OTP is correct, the adversary would now open a secure connection between his device and bank server.
- Decrypting the entire packet and dereferencing the device for the particular login id must be done within the very short timeout interval. This is what makes the attack extremely difficult to execute.

Methodology

- Designing the entire system theoretically based on requirements and threats.
- Setting up Certifying Authority for Public and Private Key Generation.
- Understanding Password splitting algorithm by A. Shamir.
- Understanding the working of multiple hashing techniques.

- Developing the basic app/GUI.
- Setting up of the bank server along with its functionality.
- Setting up multiple servers to hold the pieces of password.
- Testing each of the components of the design.

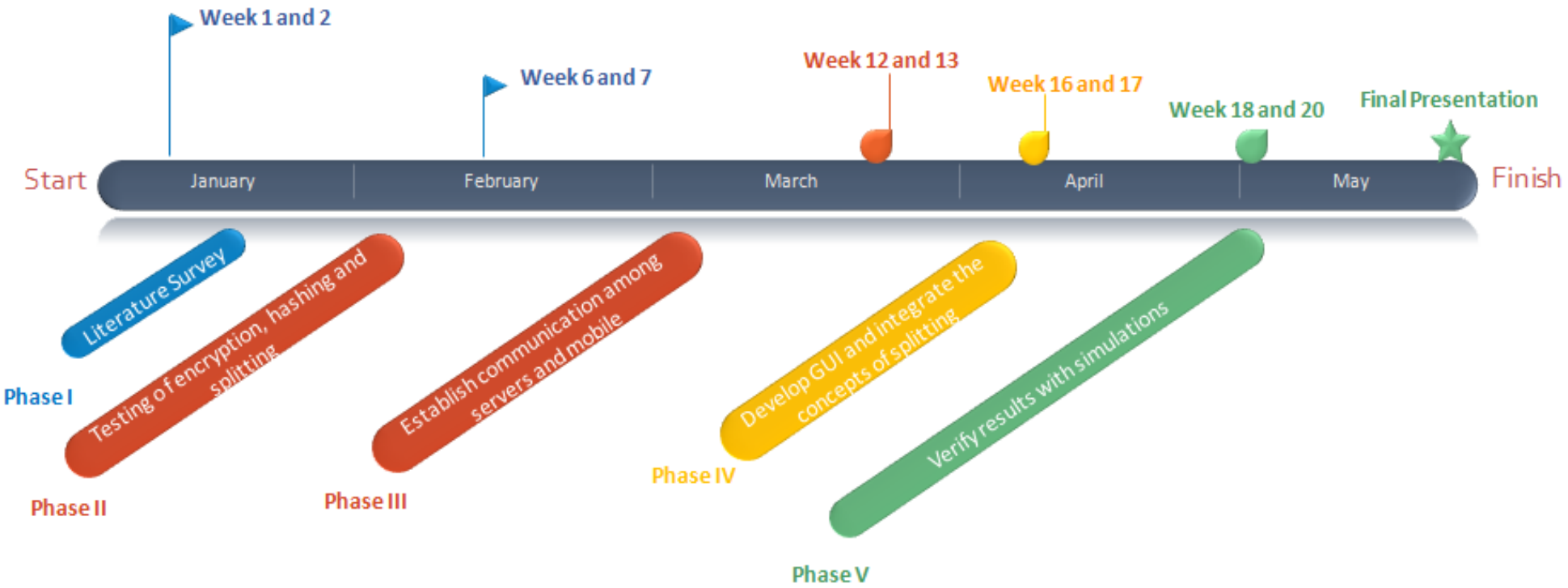
- Improving the GUI.
- Extending features of the system such as by introduction of allowing the user to change his password.
- Shuffling of the bits of the password among the servers to increase security.
- Implementing a k<n system, which makes the system resilient against corruption of any server..

Software and Hardware Requirements

Software	Hardware
Android Studio	servers/laptops
ASP .NET	Mobile
SQL	
Python/Java	

- App is going to be developed on Android
- Bank Server would involve Decryption and Splitting, hence they will be a server written in Python or Java.
- Servers storing pieces of password can be a SQL Server. This server will accept a query and return the piece for the query.

Timeline


Week 1 and 2
Week 6 and 7
Week 12 and 13
Week 16 and 17
Week 18 and 20
Final Presentation
Start
January
February
March
April
May
Finish
Literature Survey
Phase I
Testing of encryption, hashing and splitting
Phase II
Establish communication among servers and mobile
Phase III
Develop GUI and integrate the concepts of splitting
Phase IV
Verify results with simulations
Phase V

References

References

[1] HRL Laboratories, Llc. System and method for mobile proactive secret sharing. US patent: US9443089 B1, published on Sept 13th, 2016.

[2] A. Shamir. How to share a secret. *Communications of ACM, volume 22, (Nov. 1979), 612-613.*

[3] R. L. Rivest, A. Shamir and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of ACM, volume 21, (Feb. 1978).*

[4] C. L. Lui. *Introduction to Combinatorial Mathematics.* Mcgraw Hill, New York, 1978.

[5] Behrouz A. Forouzan. *Cryptography and Network Security.* Mcgraw Hill, New York, 2008.

[6] D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001.