

## **Smartphone MFA password manager**

**Jay Lohokare**

**111492930**

### **What's the problem addressed:**

Password managers are commonplace, and most browsers provide password managers as inbuilt functionality or as addons. Such password managers usually follow certain mechanism/paradigms. Most password managers save logins locally on the machine in encrypted manner. There can also be a master password that needs to be entered for accessing the login data.

Example – Firefox inbuilt password manager:

Users set a Master password, which is stored into a hash value after adding a random string and then applying SHA-1 algorithm. This generated hash value is then used to encrypt the login credentials. Firefox uses triple DES for encryption of credentials. Firefox's security flaws in its password manager are well-known (Weak SHA-1, easily accessible locations for storing encrypted data). With modern GPUs getting really powerful, SHA-1 can be easily broken. Breaking a more advanced SHA-256 hash is also possible thanks to GPUs and high speed internet. To add to this, entering passwords is often cumbersome, sharing access with others isn't possible (Without sharing the actual passwords).

The aim of this project is to build a system that eliminates security issues related to storing encrypted credentials locally, makes sharing logins easy, and uses smartphone's biometric sensors as the basis of security in password management.

### **Possible solutions:**

There are multiple password managers available as extensions of web-browsers. Typically, there are 2 approaches – Saving encrypted credentials locally or on a secure server. The master password is used to enable access to these passwords (Saved locally or on remote servers). The browser plugin detects HTML forms in the websites, and prompts users to save new passwords/auto-fill saved passwords. The key is to save domain names as primary search key. Once a particular website is detected, the plugin fetches login credentials for that website. The plugin then adds a UI layover list on the HTML form for letting user autofill the credentials. The form will then get auto-complete like feature based on credentials fetched from the encrypted storage. A new credential is detected when a HTML form is submitted and the username/password doesn't match to existing ones in the database.

## Parts of the browser plugin:

Client application that has following features:

- a. isMasterLoggedIn()
- b. masterLogin()
- c. masterLogout()
- d. heartbeat()
- e. detectForm ()
- f. fetchCredentials()
- g. loadAutofill()
- h. autoCompleteCredentials()
- i. detectNewCredential()
- j. saveNewCredentials()
- k. requestAccess()
- l. generateBarcode()

isMasterLoggedIn():

Returns True if master login is true. The master login Boolean is reset to false everytime browser is shut down. The sessionID is updated (timestamp is refreshed) using heartbeat() function. If isMasterLoggedIn() is false, masterLogin() function is triggered. This function also ensures that the session timestamp is refreshed. If the session timestamp has expired, the heartbeat() function gets a new sessionID.

masterLogin (smartphoneID):

This function triggers a master login UI. The user is asked to scan a barcode on his/her smartphone. Stores smartphone unique key (got from barcode) if smartphone application sends a success ack.

returns a session ID that's saved locally and used for all further functions.

masterLogout(smartphoneID, sessionID):

Triggered every time browser is shut or if user clicks Logout button on the UI. The default setting will be to logout when browser is shut but can be configured to not logout everytime. Reset smartphoneID and sessionID.

heartBeat(sessionID, smartphoneID):

Sends heartbeat to smartphone with session ID to indicate the web-app is alive. This function resets the session timestamp on web-app.

detectForm():

Works only if isMasterLoggedIn() returns True. Detects domain names and HTML form in a website. Triggers fetchCredentials() to get corresponding login credentials from the encrypted credentials storage.

`fetchCredentials(sessionID, smartphoneID, domain):`

Triggered if `isMasterLoggedIn()` and `detectForm()`. Sends request to smartphone to fetch login credentials for the detected website and user. Triggers `loadAutoFill()` function based on data fetched.

`loadAutoFill(listOfData):`

Uses data fetched from `fetchCredentials()` to populate a autofill feature. The autofill feature does the following – The user can start typing in the HTML form's username/name, and the function will load autocomplete of the text in a list. When user clicks on one of the elements in the list, `autoCompleteCredentials()` function is triggered.

`autoCompleteCredentials(data):`

Loads data passed by list in `loadAutoFill()` and populates the HTML form.

`detectNewCredentials():`

This function is triggered when user submits a HTML form. If the function detects new data filled in HTML form, the function triggers `saveNewCredentials()` function to save the credentials.

`saveNewCredentials(sessionID, smartphoneID, domain):`

Triggered by `detectNewCredentials()` function. This function gets credentials data and sends it to smartphone using ID of smartphone.

`requestAccess(requesterSmartphoneID, smartphoneID):`

Similar to `fetchCredentials()`, but doesn't send request to connected smartphone. Instead, the user is asked another unique smartphone identifier (Example - ID of a friend). A `fetchCredentials()` request is sent to smartphone corresponding to entered smartphone ID, to get corresponding login credentials from friends/other users.

`generateBarcode(deviceID):`

Generates barcode using browser device unique identifier.

## Parts of the Android app:

### Features in Android app

1. createSessionID()
2. sendSessionID()
3. updateSessionID()
4. signup()
5. scanBarcode()
6. notifyOnRequest()
7. grantAccess()
8. heartBeatHandler()
9. registerBioMetrics()
10. verifyBioMetrics()
11. handleNotificationClick()
12. saveNewCredentials()

createSessionID(timestamp, IMEI):

Creates a unique number based on timestamp and device IMEI

updateSessionID(oldSessionID, timestamp):

Updates the timestamp corresponding to session ID

signUp(username, password):

Creates a new profile for user on smartphone. Creates hash of username + password + salt string which will be used for encrypting credentials. Calls

scanBarcode(barcodeImage):

Uses camera app to scan the barcode. This function then triggers createSessionID() and triggers sendSessionID()

sendSessionID(sessionID, receiverID):

Sends session ID back to web-browser client (ReceiverID) when the user scans barcode on browser or opens the browser again.

notifyOnRequest(sessionID, smartphoneID, domain, receiverID):

When user requests credentials on browser (Sends request to smartphone), the smartphone generates a notification on smartphone.

heartBeatHandler(sessionID, smartphoneID):

Updates timestamp corresponding to session ID and calls sendSessionID() function using the same session ID (If not expired) or new sessionID (If expired). This acts like ACK message for requests sent by password manager clients.

registerBioMetrics(biometricsData):

Registers biometrics and generates a new hash using username + password + unique hash generated from biometrics to use it for encryption of credentials

verifyBioMetrics(biometricsData):

Verifies biometrics of users, returns True if verified.

checkSessionIDs(sessionID):

Checks if sessionID is valid

handleNotificationClick(sessionID, domain, receiverID):

Triggered when user click on request notification. Calls verifyBioMetrics() first and then calls checkSessionIDs. If True, it then calls sendCredentials() on corresponding receiverID, domain.

saveNewCredentials(sessionID, data, domain):

Calls checkSessionIDs() and then uses hash of biometric ID to encrypt the data received, save to database, with key as domain.

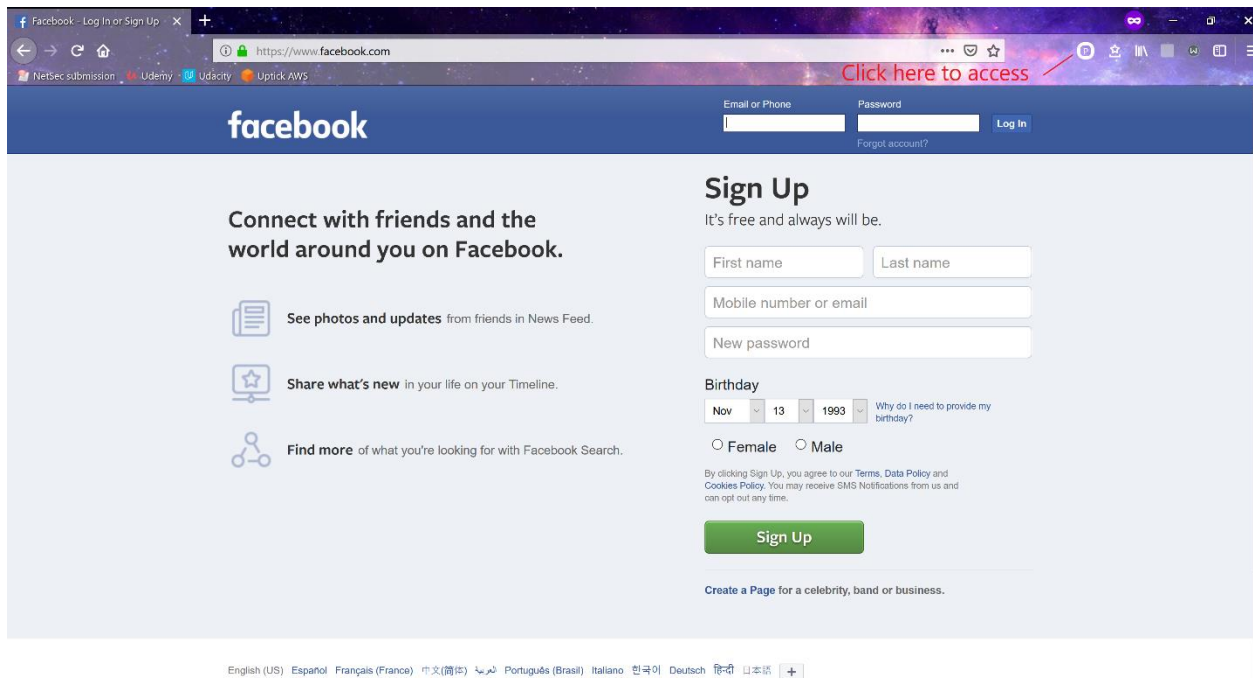
API structure:

1. End to end encrypted MQTT/XMPP protocol (Pub-sub protocol)
2. Every device has an unique ID
3. The unique IDs have identifiers to differentiate smartphone app and password manager clients
4. Browser plugin is Websockets client for the MQTT/XMPP protocol

**What does the project achieve?**

1. All standard Password manger features
2. No need to enter master passwords (Removes security issues related to theft of master password).
3. Fingerprint/FaceID is the new password (Based on smartphone). Even if phone is stolen, only biometrics can make the passwords accessible.
4. Getting access from other users is now possible – Enter smartphone ID of user whose account access is required. The user will get a notification on his/her smartphone, and can grant access using same mechanism.
5. End to end encrypted (SSL) data transfer makes data transfer between smartphone and browser secure

## Accessing the Plugin



Facebook - Log In or Sign Up

https://www.facebook.com

Click here to access

facebook

Email or Phone Password Log In

Forgot account?

**Sign Up**  
It's free and always will be.

First name Last name

Mobile number or email

New password

**Birthday**  
Nov 13 1993 Why do I need to provide my birthday?

☐ Female ☐ Male

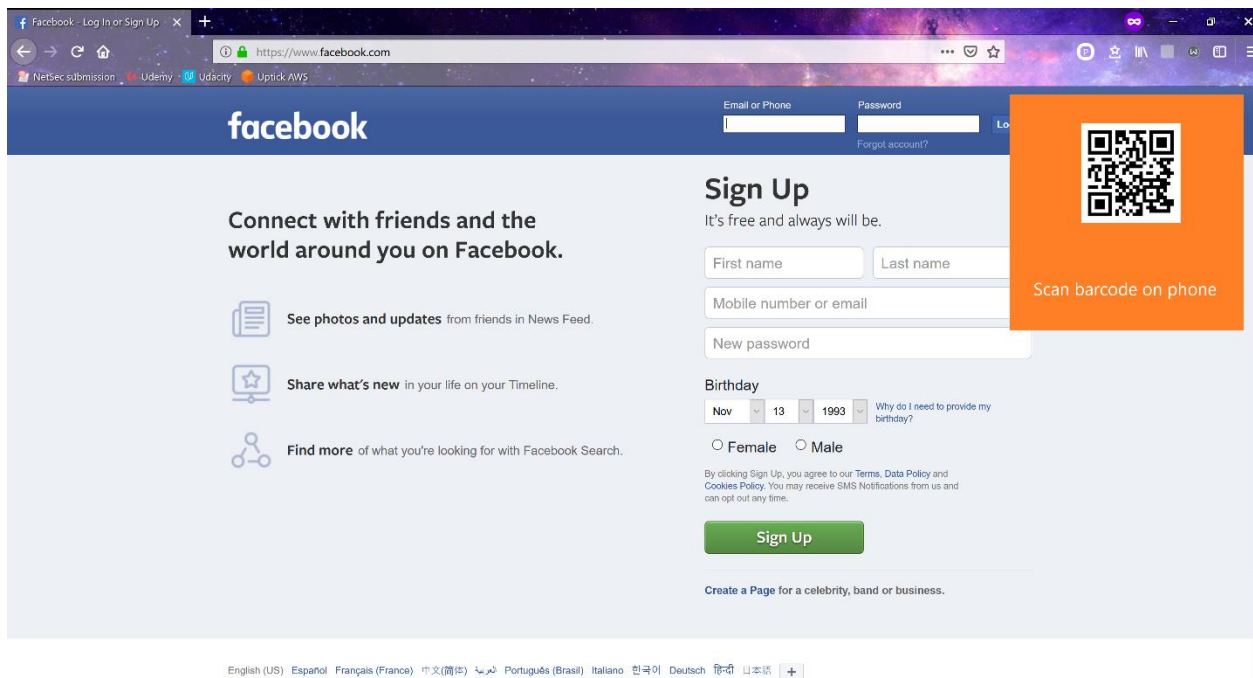
By clicking Sign Up, you agree to our Terms, Data Policy and Cookies Policy. You may receive SMS Notifications from us and can opt out any time.

**Sign Up**

Create a Page for a celebrity, band or business.

English (US) Español Français (France) 中文(简体) العربية Português (Brasil) Italiano 한국어 Deutsch हिन्दी 日本語 +

## Connect with smartphone



Facebook - Log In or Sign Up

https://www.facebook.com

facebook

Email or Phone Password Log In

Forgot account?

**Sign Up**  
It's free and always will be.

First name Last name

Mobile number or email

New password

**Birthday**  
Nov 13 1993 Why do I need to provide my birthday?

☐ Female ☐ Male

By clicking Sign Up, you agree to our Terms, Data Policy and Cookies Policy. You may receive SMS Notifications from us and can opt out any time.

**Sign Up**

Create a Page for a celebrity, band or business.

English (US) Español Français (France) 中文(简体) العربية Português (Brasil) Italiano 한국어 Deutsch हिन्दी 日本語 +

Scan barcode on phone

## Loading possible auto-complete from API

Facebook - Log In or Sign Up

https://www.facebook.com

facebook

Email or Phone Password Log In

al  
alshokare@gmail.com  
alshokare@yahoo.com

Forgot account?

### Sign Up

It's free and always will be.

First name Last name

Mobile number or email

New password

Birthday

Nov 13 1993 Why do I need to provide my birthday?

☐ Female ☐ Male

By clicking Sign Up, you agree to our Terms, Data Policy and Cookies Policy. You may receive SMS Notifications from us and can opt out any time.

Sign Up

Create a Page for a celebrity, band or business.

English (US) Español Français (France) 中文(简体) العربية Português (Brasil) Italiano 한국어 Deutsch हिन्दी 日本語 +

## Requesting access to other users

Facebook - Log In or Sign Up

https://www.facebook.com

facebook

Email or Phone Password Log In

Forgot account?

### Sign Up

It's free and always will be.

First name Last name

Mobile number or email

New password

Birthday

Nov 13 1993 Why do I need to provide my birthday?

☐ Female ☐ Male

By clicking Sign Up, you agree to our Terms, Data Policy and Cookies Policy. You may receive SMS Notifications from us and can opt out any time.

Sign Up

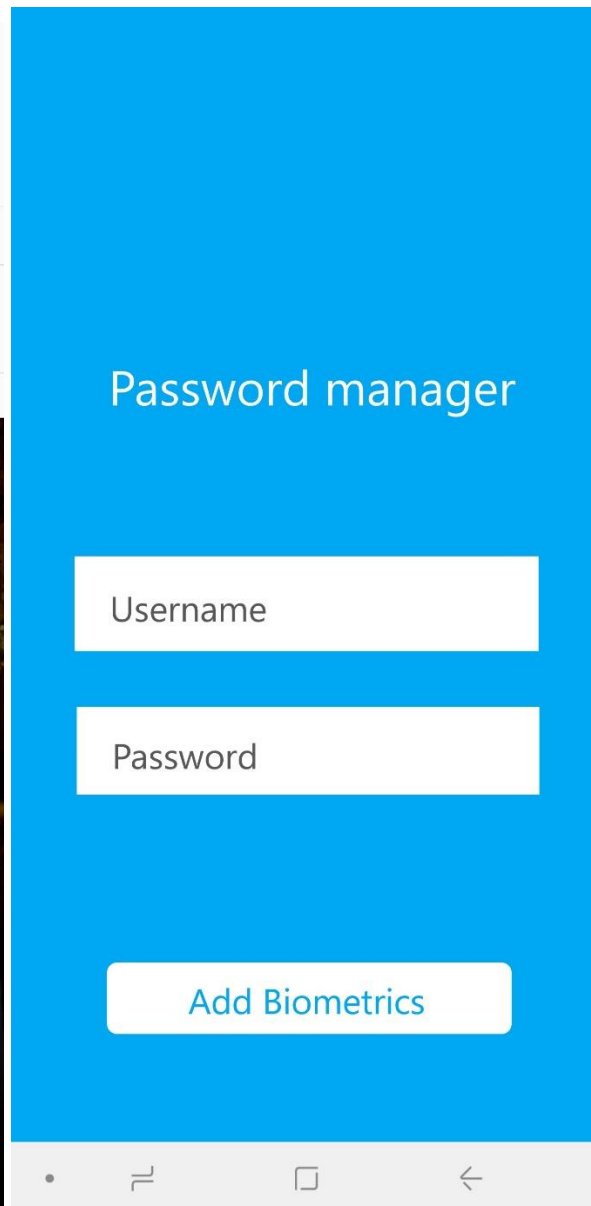
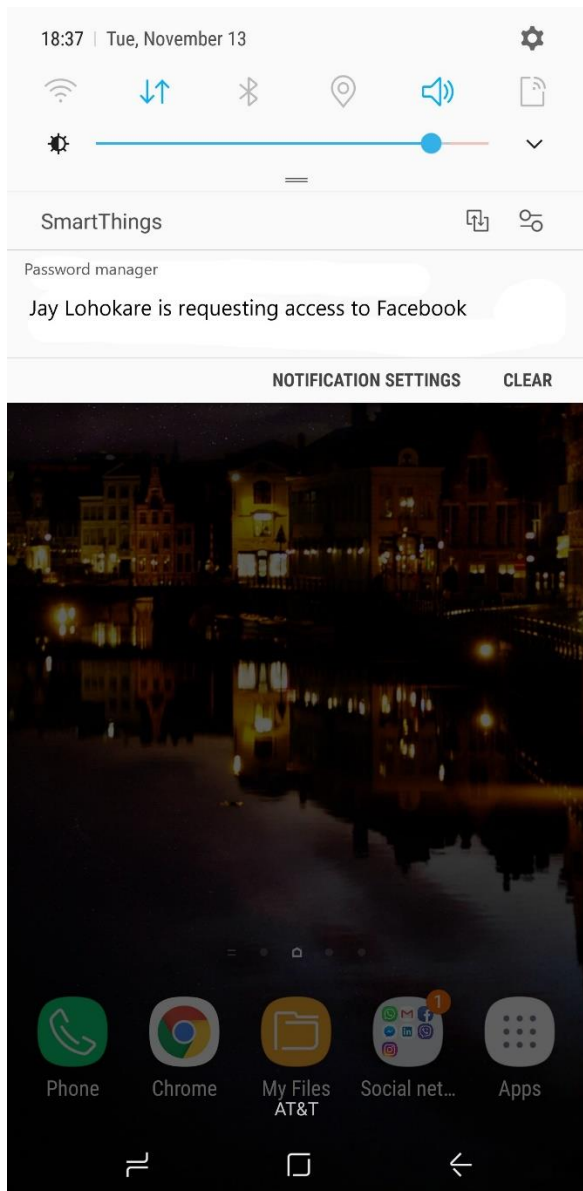
Create a Page for a celebrity, band or business.

English (US) Español Français (France) 中文(简体) العربية Português (Brasil) Italiano 한국어 Deutsch हिन्दी 日本語 +

Request access

Enter smartphone ID



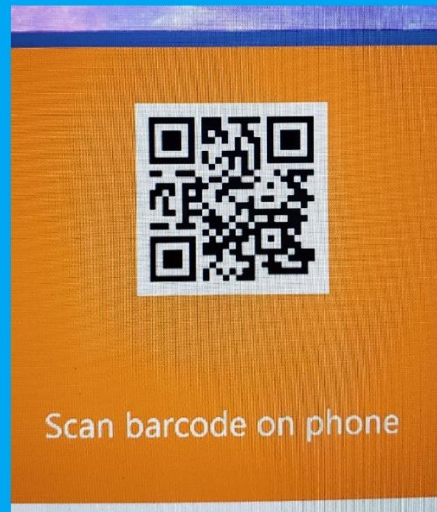


Verify  
Biometrics

Verified

Error

Scan  
Barcode



Detected barcode

