ISSN: 0974-2115
www.jchps.com Journal of Chemical and Pharmaceutical Sciences

MobiAuth: A new way for mobile authentication & authorization into third party websites

J Jeyasudha, S Krishnaveni, Jothi B
Department of Software Engineering, SRM University, India.
*** Corresponding author: E-Mail: jeyasudha.j@ktr.srmuniv.ac.in**

ABSTRACT

End Users logs into third party applications or web sites using their Microsoft, Google, Face book, Twitter, One Network etc. accounts without providing the passwords. OAuth provides to clients a secure usage of server resources on behalf of the resource owner. To give authorization to third party access to their server resources, not sharing their credentials a process is specified that will work with Hypertext Transfer Protocol Secure (HTTPS). The access tokens issued to clients by the authorization server approved by resource owner to access the resources. OAuth (Ping Identity, 2015) despite its advantages failed to prevent several breaches and brute force attacks that can and have broken the schema. MobiAuth proposes the utilization of fingerprint scanners available in mobile phones to locally authorize the user and send the IMEI number available locally in the phone to the resource owner to authenticate and authorize the user. In turn the client application will receive tokens generated and based on which the client application can authenticate the user. In the mobile platform the IMEI/MAC number is a unique number which can be used for authentication and authorization.

KEY WORDS: Authentication, Authorization, OAuth, IMEI, MAC Number

1. INTRODUCTION

Many authentication schemes depend on secret passwords. Unfortunately, the length and randomness of user-chosen passwords remain fixed over time. In contrast, hardware improvements constantly gain attackers increasing computational power. As a result, password schemes such as the traditional UNIX user-authenticated systems are failing with time. Password recovery by brute force attacks can be affected by effected based on the type of system, length of the string as well as the range of characters allowed.

Background

OAuth: IETF proposed a creation of a standard to facilitate communication between two independent websites involving the exchange of information based on authentication. This standard was possible by the exchange of tokens (Ping Identity, 2015). The user would be authenticated by the resource owner and the web application needing the resource would be given a pair of tokens based on which the user can be authenticated without the re-entry of his/her manual authentication. OAuth despite its advantages failed to prevent several breaches and brute force attacks that can and have broken the schema. MobiAuth proposes the utilization of fingerprint scanners available in mobile phones to locally authorize the user and send the imei number available locally in the phone to the resource owner to authenticate and authorize the user. In turn the client application will receive tokens generated and based on which the client application can authenticate the user.

Two-Factor Authentication (2FA): Two-factor authentication is a security process in which the user provides two means of identification from separate categories of credentials; one is typically a physical token, such as a card, and the other is typically something memorized, such as a security code. In this context, the two factors involved are the user's mobile device that allows for a local authentication based on biometrics, namely the user's fingerprint and further an OAuth based authorization schema. This allows the system to promote a pseudo Three-factor authentication schema in addition to the true two-factor authentication that is already in place.

MobiAuth: MobiAuth is required for Delegating Access to Certain Party, for Certain Resource, For Limited Time and can be selectively be revoked. The server hosts the protected resources using access tokens.

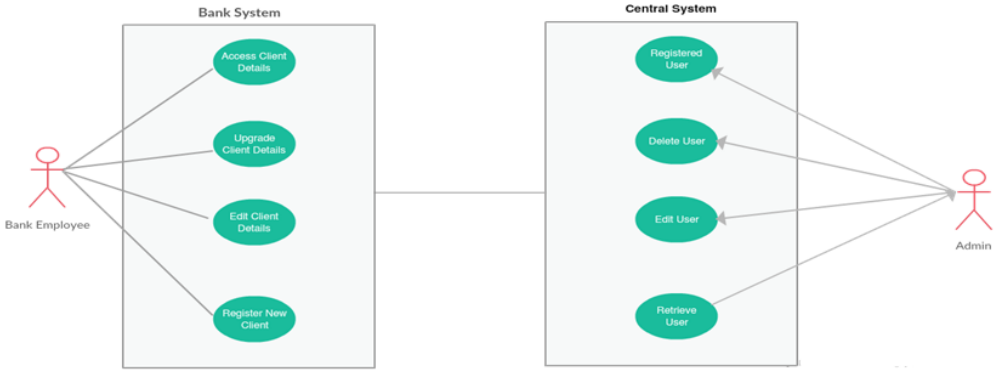

Figure.1.MobiAuth in Banking System

AAA of Password Security: Authentication (& Identification) -Establishes that the user is who they say they are (credentials). Authorization -If the authenticated person is allowed to access specific information or functions.

Access Control -Restriction of access (includes authentication & authorization) Example: Passwords (Provos, 1999) are Stored in Filing System as Clear text, in Dedicated Authentication Server as Clear text, and then Encrypted, Hashed and Salted Hash. Passwords are NOT truly random. It can be 52 upper/lowercase letters, 10 digits, and 32 punctuation symbols equals ≈ 6 quadrillion possible 8-character passwords. Different systems impose different password requirements. Passwords have to be changed often. Some passwords are only used occasionally.

2. MATERIALS AND METHODS

Proposed Architecture of Mobiauth: The architecture consists of essential of two independent systems that communicate between each other. The Client consists of native users and non-native users whose data needs to be accessed from an external system. The client is required to store the external systems data and other information that is transferred. The external application holds the information that is required by the Client whose requesting the information. (Hardt, 2012) The resource owner has several restrictions in place in transfer of the data due to their native client's security and data confidentiality concerns. The authentication server has three logical flows. The first involves a new user who has never accessed the system before and is doing so by entering his credentials based on which if he is authenticated then the system issues temporary tokens and refresh tokens. If the user is a returning user , then he is request to share his or her temporary token to the server and based on which he is authenticated , if this fails he is verified on basis of his refresh token and if successfully authenticated on basis of refresh token, then the server re-issues his tokens for further authentications. If both the tokens are incorrect then the user is made to sign in on basis of credentials.

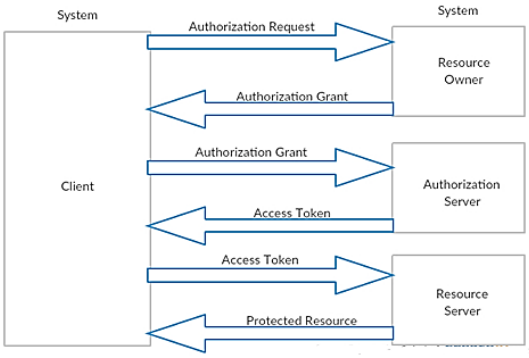

Figure.2. Architecture of MobiAuth

If the user is authenticated, then on basis of permission bits that are recovered based on the client ID of the system of the client request access, we will be able to retrieve the correct fields and transfer them to the external application requesting them from the resource owner

Client Side Web Application: A single database that stores the client id, client password, usernames, the fields that of interest to the application for persistence, refresh and access token. The user choose to login via MobiAuth and then it redirects to the central server.

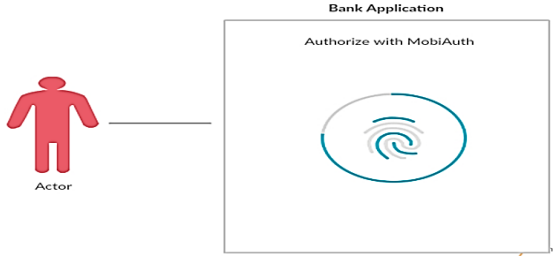

Figure.3. Client Side Web Applications

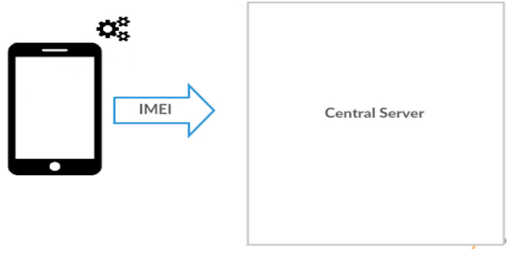

Figure.4. MobiAuth with the Central Server

A mobile phone equipped with fingerprint scanner. The phone first enters the central server by using web view(in case of android) and then is able to authenticate the user based on his/her fingerprint. When the user is authenticated the mobile sends along with redirected url the phones IMEI/MAC number.

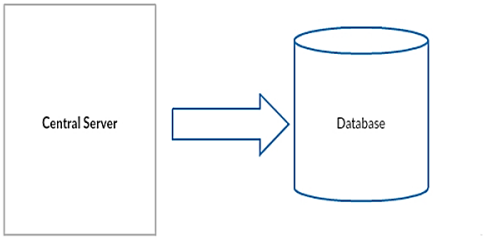

Figure.5. Central Application Module

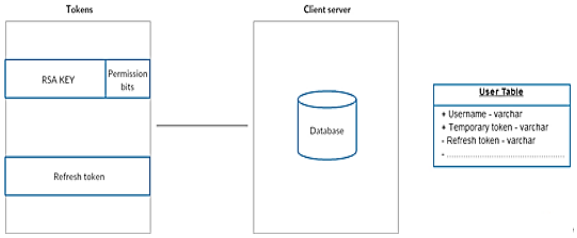

Figure.6. Token Generator of MobiAuth

When the IMEI number comes to the central server from the mobile module it checks if the user is genuine and forwards the refresh token and access token to the client application. A hash generator and a client id recognizer.

The tokens are generated using an encryption algorithm (bcrpyt) and the tokens are stored both on the client and central server. In addition to the generated hash the permission bits are generated based on the client ID. The hash values are repeatedly refreshed on the central server in order to prevent unsolicited access.

Implementation of Mobiauth: MobiAuth can be implemented with following modules Client side web application for giving the request and IMEI number, Central Application Module is for the authenication, Token Generator of MobiAuth for generating the session key for the website or their resources they want to access In the MobiAuth, the following steps will be performed

Step 1: User registers with MobiAuth and as a result the login gets stored in MobiAuth

Step 2: The person who wishes to access the database of MobiAuth registers with MobiAuth to get his client id and client pass along with permission bits if successful gets redirected.

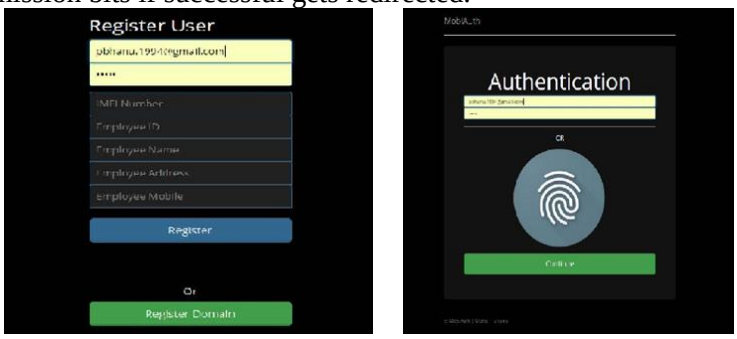

Figure.7.Registration & Authentication Page of MobiAuth

Step 3: User going to login to the app that wishes to use MobiAuth's database

(Database is sent as authorization of the applications credentials MobiAuth page verifies the details of the application and if successful sends)

Step 4: User sends to MobiAuth login page

(if user first time login then generates tokens and shared across to application)

Step 5: if user successful then gets information sent to the application that needed to access MobiAuth information

3. RESULTS ANALYSIS AND DISCUSSION

Based on the type of attack we decide to utilize, we must provide the toolkit with the hash values. Hash values depend on the string that is encrypted. In order to determine the resistance of MobiAuth's hash values against non-mobiauth generated hash values that are derived usually based on user's strings, we will conduct the experiment with a set of hash values of both the types.

Attack on Non-Mobiauth Based Hashvalue: In order to carry out these attacks we take a set of user strings and enter them into the Md5 Calculator that generates the corresponding md5 hash. These hashes are later entered into the toolkit. Based our constraints we deploy a suitable wordlist.

Attack on Mobiauth Based Hashvalue: Similar to the last test we will enter in the hash value, but instead be using the pre hashed value generated by our system that is stored in our MobiAuth database. An the results after the attack can be valued based upon the hash valu.

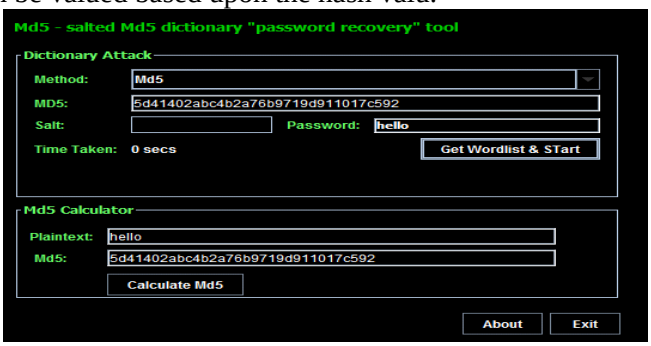

Figure.8.Non-MobiAuth Based hash value attacked

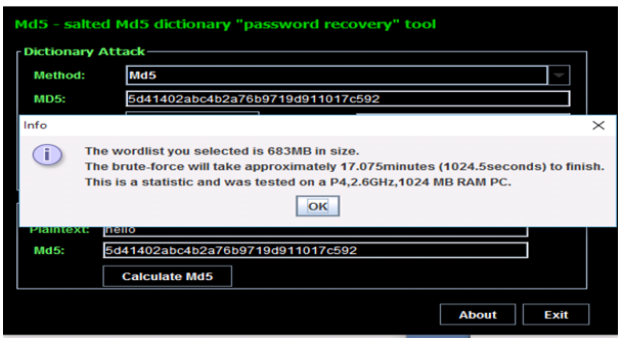

Figure.9. Selection of Attack

Based on the results gathered by the attacks we can project the time it takes in minutes for MobiAuth's hash values to be broken as a factor dependent on the character sets limits and class of the attack. Based on the results the following can be concluded:

Higher Character sets and lower class of attacks take the highest time to crack.

Higher Character sets and higher class of attacks takes higher time to crack.

Lower Character sets and lower class of attacks take lower time to crack.

Lower Character set and higher class of attacks takes the lowest time to crack.

Figure.10.Result post attack

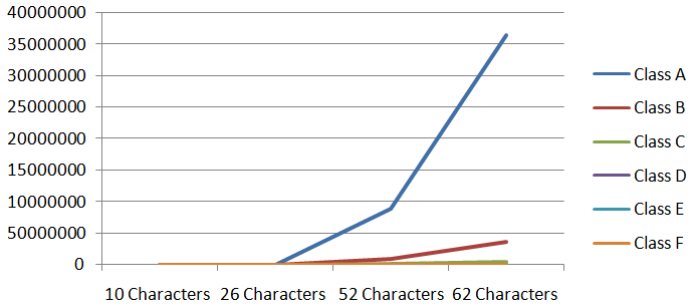

Figure.11.Password recovery

Extrapolating the information it is observed that MobiAuth's schema for generating and recycling hashes allow for a best case scenario of 20 minutes and a worst case scenario of 5 minutes under which only certain character sets are able to be cracked and as a result of which are excluded in implementation.

4. CONCLUSION

In the existing mechanisms, the problem is to give credentials of the user with the help of the social networks. It ultimately leads to security issue. To avoid that MobiAuth is used to provide security with IMEI/MAC address. And to provide more security with the help of the wireless connection access and to limit the range of the End User. taken into consideration the implicit constraints imposed upon the system as a result of using local authentication biometric systems such as fingerprint scanners provided in the mobile device platforms such as Android and IOS. In spite of these hurdles our system is able to maintain a highly scalable and preserve customer data confidentiality by incorporating a highly randomized string that is encrypted and varies explicitly with time.

REFERENCES

Hardt D, The OAuth 2.0 Authorization Framework, Internet Engineering Task Force, 2012, 2070-1721.

Niels Provos and David Mazières A Future-Adaptable Password Scheme. Proceedings of the FREENIX Track, USENIX Annual Technical Conference, 1999.

Ping Identity, The Essential O Auth Primer: Understanding O Auth for Securing Cloud APIs, 2015.

Provos N & Mazières D, A Future-Adaptable Password Scheme, USENIX Annual Technical Conference, 1999.

She W & Thuraisingham B, Security for Enterprise Resource Planning System, Information Systems Security, Informa UK Limited, 16, 2007, 152-163.