International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

Multi-factor Authentication for Bank Using Mobile Phones

Jayshri salunke[1], Jagrutisonar[2], Komal sonawane[3]

[1]*Jayshri salunke, Shivajirao s.Jondhle College of Engineering*
[2]*jagruti sonar, Shivajirao s.Jondhle College of Engineering*
[3]*Komal sonawane, Shivajirao s.Jondhle College of Engineering*
[4]***Dr. k.k.Tripathi**, Dept. of computer Engineering, Shivajirao s.jondhle College of Engineering, Maharastra India,*

Abstract - *As computing becomes dominant and more intensive, people depend on public computers to do transactions and business over the World Wide Web. Due to the penetration of technology, Internet has become the environment of choice for a variety of services via electronic means: ebusiness, e-commerce, e-banking, e-voting etc. Financial institutions and organizations providing Internet based products and services to their customers and end users should use effective and secure methods to authenticate the identity of customers using those products and services. Multi-factor authentication provides improved level of security and delivers an authentication assurance for sensitive transactions. Traditional method of generating and delivering OTP (One Time Passwords) messages are generally relayed via SMS channel. Depending on the area of operation and provider, international roaming, SMS costs and delays put restrictions on this existing system. Our projected system presents a multifactor authentication procedure in which a user's device produces an OTP from an initial seed consisting of unique parameters. This proposed system overcomes the restrictions by the SMS system.*

Keywords *– user, client, server, seed, otp, md5, sha-256, chain hashing, hash collision*

I. INTRODUCTION

1.1 System of Authentication

It is the process of determining whether someone or something is, in fact, who or what it is declared to be. Logically, authentication precedes authorization. The two terms are often used in combination but they are in fact two different processes.

1.2 Authentication vs. Authorization

Authentication is a process of comparing the provided credentials during logging in with respect to the file contained in the user database of the OS or within a server. The process is completed and the user is authorized for accessing system after matching the credentials. According to the authentication level, the environment the user can see, the method of interaction and the amount of allocated storage space are defined by the permission and folders.

In case of authorization the admin or the super user does the job of granting rights and checks the permission of the user account for accessing the resources. The user's permissions are stored locally or on the server and define the privileges and preferences granted for the authorized account. The administrator sets the settings which are defined by environment variables.

1.3 Traditional System of Authentication

Existing system of authenticating and authorizing users is generally carried out using One-factor Authentication (1FA) or Two-factor Authentication (2FA). 1FA accepts 2 fields, an Email-ID/Phone

Number/Username and a password. 2FA uses the same fields as 1FA but prompts the user to enter OTP sent to the mobile phone. The latter is more secure than the former as it adds extra level of barrier for logging into websites. However, 2FA is generally carried out using SMS channel which is insecure and prone to attacks. In our case, Online banking requires strong user authentication.

User authentication is often achieved by utilizing a one-factor or two-factor authentication technique based on something the user knows, i.e., a static password, and something the user has, i.e., an OTP. The major advantage of involving a mobile phone is that most users already have mobile phones, and therefore no extra hardware token needs to be bought, deployed, or supported. The traditional 2FA system works by sending an OTP over an SMS to a user who wants to make an online transaction.

1.4 Drawbacks of existing system of Authentication

One-factor Authentication (1FA)

Single point of failure during login process

Limited to known or managed laptops/desktops

Need to monitor application upgrades and changes

Lowest level of security

Two-factor Authentication (2FA) (SMS-based)

SMS delay due to network congestion, operator outages and network connectivity

Unavailability of service due to low signal strength and coverage area

Unavailability of device as user needs to have the device physically present during login process 4) Regulatory restriction which block bulk SMS gateways

II. BACKGROUND

Leslie Lamport [1] was the first to conceptualize the idea of OTP in the early 1980s. His principle stated that each time a user logs into the system, a pre-defined algorithm generates a pseudorandom output which increases the security. An OTP is a single-use password which is valid only for a single login transaction or session.

2.1. The S/Key OTP System [2]

It uses a computation technique to generate a finite sequence of single passwords for one time usage from one secret "seed." The security is completely dependent on this secret seed. Only the user knows about the secret seed. The single-use passwords are dependent on each other such that it makes it computationally difficult to compute any password from the initial sequence. A hash function h (.) is applied for "n" times to a seed "s" forming a hash chain with a length "N". This method is valid only to a certain iterations of authentications N, so that after reaching N authentications, a full process cycle restart is needed.

2.2. Bicakci et al.'s Scheme [3]

The infinite length hash chains (ILHC) use a public-key algorithm for operation. It produces a function which is infinite and forward in nature called one-way function (OWF.) This OWF is the core of OTP generation. Bicakci et al. proposed a protocol using RSA algorithm, in which e is the public key and "d" is the private key. Using the RSA public-key algorithm, the OTP which originates from initial input "s" for the tth authentication is as follows: OTPt (s) =At (s, d)

The verification of the tth OTP is done by the process of decryption. OTPt (s) using e is as follows:

A (OTPt (s), e) = OTPt-1 (s)

The computational complexity increases as we increase the number of cascaded exponentiations, making this algorithm very difficult to implement in devices having limited processing power Example: Mobile Phones

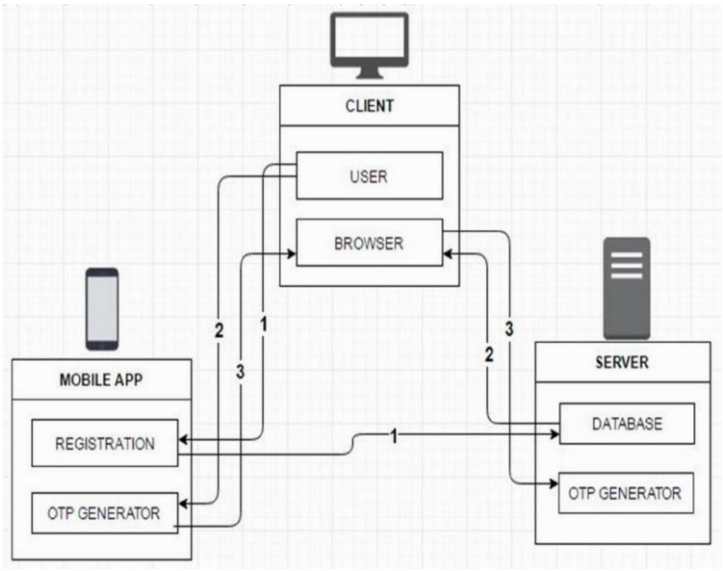
2.3. RSA SecurID Authenticator [4]

It uses a hardware based or a software based token, which could be hardware or software aided by a main server synchronized internal clock. Each token is uploaded with a unique seed, from which a pseudo-random number is generated. Every 60 seconds an OTP is generated using this token. A mathematical operation is performed considering the current timestamp and the loaded seed. The same process occurs in synchronization and in parallel at the server side. User authenticates himself using the generated OTP with his corresponding PIN. Such synchronization is not possible in the case of mobile phones.

III. OUR APPROACH

This system is an extension of Lamport's [1] idea with some minor changes in order to produce infiniteness and forwardness. We have tried avoiding the use of public key cryptography. Using two different one-way hash functions we need to integrate Lamport's methodology.

3.1 Multi-factor Authentication Architecture



1. The user registers and logs in using credentials to the bank's website, requesting access. As a response to this access request, a SSL session is established.

2. The server randomly challenges the user or the client with new indexe variables. The user enters the same indexes in the Android app to get the corresponding client sided OTP.

3. The user responds the OTP. The server compares the client sided OTP with its own calculated server sided OTP.

3.2 Multi-Factor Authentication Entities

4. Every entity is divided into sub-parts according to the functional modules.

5. 1. Android Mobile App: The interface with which user registers and generates OTP.

   A) OTP Generator: Standalone function, No Internet connection needed. - Generates seed by concatenating IMEI, IMSI and A/C Number

6. - Generates corresponding OTP from client seed by chain hashing.

7. B) Registration: Bank's website rendered inside mobile, Internet connection needed. - Sends IMEI, IMSI to the server database.

8. - Registers a new user assigning him a unique UserID and A/C Number.

9. 2. Client: The end user-machine system interacting with the server. A) User: The human interacting with the system.

10. Does registration, logging in, answering security question etc.

11. Performs email-id confirmation, account recovery operations etc.

12. B) Browser: Desktop or Android based, Example: Chrome, Firefox etc.

13. For logging into the system with proper credentials.

14. To enter client sided OTP for comparison.

15.

16. 3. Server: Linux based Remote Web Hosting

17. A) Database

18. - For data processing, storage and modifications. - For OTP checking and comparison.

19. B) OTP Generator

20. - Generates seed by concatenating IMEI, IMSI and A/C Number - Generates corresponding OTP from server seed by chain hashing.

3.3 Login and registration on phases registarion

1. User registers on the bank's website via mobile app providing the following details as input - Unique Username and Email-ID

21. Desired Password

22. Security Question and Corresponding Answer

23. Using Android's "android.telephony.TelephonyManager.getDeviceId( )." permission, the mobile app reads and sends the IMEI and IMSI number to the server.

24. Server receives all the input data and creates an account for the user with following parameters

25. Unique Username bound to the Email-ID

26. User ID associated with Bank Account Number

27. Security Question and Corresponding Answer

28. Server will display the above input data on the mobile app which was used to create an account.

29. Server will concatenate IMEI, IMSI, Bank A/C Number to generate a seed. Server → Concatenate (IMEI, IMSI, A/C No) → Server Sided Seed

Registration: Phase II

30. User will now have to enter his newly created Bank A/C number in the OTP section of the mobile app.

31. The mobile app will concatenate IMEI, IMSI, Bank A/C Number to generate a client seed. Mobile App → Concatenate (IMEI, IMSI, A/C No) → Client Sided Seed

Login: Phase I

32. User will now request login into bank's website using username and password

33. Server will validate the input data and create a secure SSL session.

34. Server will create and display 2 random index variables (x, y) and send them to the client 4. The server will use the same variables to generate a server sided OTP from seed using nested hashing functions or a hash chain.

35. Server Sided Seed → SHA(Seed) → MD5y(SHAx(Seed)) → Server Sided OTP

Login: Phase II

1. User will input the server generated 2 index variables (x, y) in the mobile app's OTP Section.

2. The mobile app will generate a client sided OTP using nested hashing functions from the seed.

3. Client Sided Seed → SHA(Seed) → MD5y (SHAx(Seed)) → Client Sided OTP 4. User will now enter the client sided OTP in the bank's website.

5. Server will now compare its own generated OTP and client sided OTP.

Login: Phase III

36. If equal, the user is prompted for a security question from the server.

37. The user inputs the correct answer which is case-sensitive.

38. The server checks the input data by verifying it in the database.

39. If the answer is successfully validated, the user is fully authenticated into the system.

IV. NUMERICAL ILLUSTRATION

40. IMEI: 823589069458989, IMSI: 405151001385048, ACNO: 133701

41. Concatenate (IMEI, IMSI, ACNO)

42. Concatenate (823589069458989 + 405151001385048 + 133701)

43. Seed: 8235890694589894051510013850481337O1

44. Hashing Functions: HA = SHA256, HB = MD5

45. Hashing Index (x, y): x = Rounds for SHA256, y = Rounds for MD5 Hashing Index Limit: Min = (1,1), Max = (10,10) Formula:

46. Seed > HBy (HAx(Seed)) > Hash Value > OTP

47. i.e. Seed > SHA256(Seed) > MD5(SHA256(Seed)) > Hash Value > OTP

48. Example: index (x, y) = (1,1)

49. Formula: Seed > HB1(HA1(Seed)) > Hash Value > OTP

50. Seed > SHA-256(Seed) > MD5(SHA-256(Seed)) > 7ff023ff6e48e10b5abeff9f4169ee54 Final Hash Value: 7ff023ff6e48e10b5abeff9f4169ee54

51. Example: index (x, y) = (2,3)

52. Formula: Seed > HB2(HA3(Seed)) > Hash Value > OTP

53. Seed > HA3(Seed): SHA-256 3 times on (Seed)

54. Seed: 8235890694589894051510013850481337O1

55. 1st Iteration SHA256: 3f4d2e67a1010fd1a2072033a5f7495f9331eca9a999f8717f67b2699b31965d

56. 2nd Iteration SHA256: 582504e52b842be08c704df0b49ef6c0309d06ba193c78f7b27ed10f8dee04b4

57. 3rd Iteration SHA256: d84738654926d1ea17b2c76a9eef604bd5fc8ec0edb867928c8a8b9ec8eac0fe

58. Seed > HB2(SHA256-3): 2 times MD5 on (3rd Iteration of SHA256)

59. 1st Iteration of MD5: 3265b97837c9ab2c77c8fdf099e2cc02

60. 2nd Iteration of MD5: 812961c4ea21d22f3833833f778ff26a

61. Final Hash Value: 812961c4ea21d22f3833833f778ff26a

V. RESULTS

62. The proposed "Multi-factor Authentication" system has been analyzed from a security point of view and the following results have emerged.

63. It provides more layers of security to bypass as compared to 1FA and 2FA.

64. The data is less prone to snooping and Man-In-The-Middle attacks.

65. The system makes use of nested hash chaining, overcomes the problem of hashing collision.

66. The mobile app is fully independent of SMS channel and internet channel for OTP generation.

67. The system can resist offline guessing because it uses strong passwords and hash functions.

68. This scheme uses forward hashing techniques which eliminates small challenge attack completely.

69. Since the OTP is valid for a single login session it evades replay attack vulnerability.

VI. CONCLUSION

As mobile phones are becoming more powerful and even more cheaper these days, we have proposed a system that takes advantage of increased processing power and the availability of the phone. Using two chain hashing functions we generate a OTP from an initial seed We have defined our approach to an online authentication process involving a banking website. Our algorithm doesn't require TOTP based system. Our system overcomes the problems with utilizing OTPs with an SMS, consisting of the SMS cost and delay, along with international roaming restrictions like. The multi factor authentication property has been successful without any restrictions.

REFERENCES

1. L. Lamport, "Password Authentication with Insecure Communication", In: Comm. ACM, vol. 24, No 11, pp. 770772, 1981

2. N. Haller, "The S/KEY One–Time Password System. In: Proceedings of the ISOC Symposium on Network and Distributed System Security", pp. 151-157, 1994

3. K. Bicakci N. Baykal, "Infinite length hash chains and their applications" In: Proceedings of 1st IEEE Int.

Workshops on Enabling Technologies: Infrastructure for Collaborating Enterprises WETICE'02, pp. 57-61, 2002.

4. http://www.rsa.com/node.aspx?id=1156. [Accessed: October 04, 2010].

5. A. Menezes, P. Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, Inc. 1997

6. M.H. Eldefrawy, M.K., Khan, K. Alghathbar, E.-S. Cho "Broadcast Authentication for Wireless Sensor Networks Using Nested Hashing and the Chinese Remainder Theorem", Sensors, 10(9): 2010, pp. 8683-8695.

7. A. Chefranov, "One–Time Password Authentication with Infinite Hash Chains. Novel Algorithms and Techniques", In: Telecommunications, Automation and Industrial Electronics, pp. 283-286, 2008

8. L. Raddum, Nestås, K. Hole, "Security Analysis of Mobile Phones Used as OTP Generators", In: IFIP International Federation for Information Processing. 2010, pp. 324-331, 2010