Multi-Factor Authentication Using Threshold Cryptography

Vishnu Venukumar
Department of Computer Science and Engineering
National Institute of Technology
Calicut, Kerala 673601
Email: vishnuvp@ymail.com

Vinod Pathari
Department of Computer Science and Engineering
National Institute of Technology
Calicut, Kerala 673601
Email: pathari@nitc.ac.in

***Abstract*—Multi-Factor Authentication is used as a foolproof solution to various issues involved in present day critical authentication systems. However, it comes with the overhead of employing multiple authentication programs to complete the process. Moreover, current multi-factor authentication schemes require all intermediate One Time Passwords(OTPs) to be stored for the lifetime of the authentication process. They also involve security risks whenever an authentication process requires the user's password at a public place like a Point-of-Sale terminal or an open ATM booth. This work proposes a more secure, efficient, convenient and flexible multi-factor authentication technique using threshold cryptography.**

***Index Terms*—Threshold cryptography, Information security, Computer security, Cryptography, Authentication, Multi-factor authentication**

I. Introduction

Authentication is the process of verifying the claimed identity of an entity. The process involves the verification of one or more of some knowledge (like a password), something in possession (like a RFID access card) or inherence (like the fingerprint) of the entity.

A basic authentication process verifies anyone of the above information. Multi-Factor Authentication(MFA) is a form of strong authentication which uses more than one information among the above [1]. Multi-factor authentication techniques employ multiple authentication programs to confirm the identity of the user and eliminate the risk of compromise of secret or token in basic authentication methods. Each authentication program is a basic authentication process. Hence, the overhead involved in a multi-factor authentication process is multiple times that of a basic authentication process. This makes the process less flexible.

Present day security measures for electronic transactions employ multi-factor authentication and require users to key in no less than one token to authenticate a transaction. The token can be the user's Personal Identification Number (PIN), or a password or a One Time Password (OTP). Some sophisticated systems use biometric authentication. Nevertheless, for transactions made at a Point-of-Sale (PoS), keying in the user's PIN stands as the common choice. The common procedure to authenticate a transaction at a PoS terminal is to hand over the credit or debit card to the shopkeeper and key in the PIN when the person asks to do so. It may be followed by a signature on the transaction slip, and the whole process gets complete in about a minute or two without any frills of counting or balance calculation. Though the process is attractive in terms of convenience, it involves obvious security issues. In a typical supermarket or restaurant scenario in India, many people including the shopkeeper shoulder surf when the user keys in the PIN. The problem is further elevated to new levels when the PoS terminal device is not present at the counter, and the user should either reveal the PIN to the shopkeeper or accompany him to the remote device. This is a common sight in restaurants where users hand over the credit or debit card to the waiter and either tells him the PIN or hands him over a napkin with the PIN written on it. As far as the shopkeeper and the people standing around the user are prudent, there is no risk and relying on this false security is a mere absurdity which happens currently in millions of PoS transactions every day.

Apart from the security issues from the social perspective, the current multi-factor authentication techniques possess a few performance and efficiency concerns. All of the OTP used in each level of authentication are to be stored in the database. The number of database reads and writes increases proportionally to the number of levels used in the process. Also, the number of comparisons made to the stored OTP or secrets rise accordingly.

This work proposes a new multi-factor authentication process using threshold cryptography to make the authentication process more secure, efficient, convenient, and flexible. Threshold Cryptography is a cryptographic technique in which a secret is split into a number of shares and the reconstruction of the secret happens only when a threshold number of shares are received back [2]. A threshold cryptographic scheme is usually denoted as a $t(t, n)$ scheme, where n represents the number of shares generated and t is the threshold number of shares needed to reconstruct a secret [3]. Fundamentally, threshold cryptography deals with secure sharing of a secret with multiple stakeholders or shareholders. The first threshold secret sharing scheme was proposed by Adi Shamir and George Blakley independently in 1979. Shamir's work serves as the basis for most of the secret sharing schemes proposed till date.

The rest of the document is organized as follows. The next

978-1-5090-2029-4/16/$31.00 @2016 IEEE

section enlists a few major work in the area of threshold cryptography and multi-factor authentication. Section III explains Shamir's Secret Sharing Scheme. Section IV details the proposed multi-factor authentication technique. Section V gives an example application which uses the proposed technique. In the last section, we enlist the advantages of the proposed scheme and concludes the paper suggesting future works.

II. Literature Review

Adi Shamir introduced Shamir's Secret Sharing scheme in his seminal work 'How to share a secret?' [4] in 1979. It remains as the base for almost all the important threshold cryptographic applications. The reason for the wide adoption of Shamir's technique can be attributed to its simplicity and space efficiency. In Shamir's scheme, secret and shares are considered as points on a curve defined by a t degree polynomial where t is the threshold. While the secret shares can be any points on the curve, the secret is the y-intercept of the curve. As at least t points are required to identify the curve uniquely, it can be constructed only if more than t number of shares are known. Shamir used Lagrange Polynomial interpolation to reconstruct the curve which effectively yields the rest of the points on the curve. The secret is regained by finding the y-intercept of the curve. A detailed description of the scheme is provided in Section III.

George Blakley also published his work on secret sharing in 1979 in his paper titled 'Safeguarding Cryptographic Keys' [5]. Blakley used the intersection property of hyperplanes in a t-dimensional space to devise his technique. It relies on the fact that any t $(t-1)$ dimensional hyperplanes intersect at a point. The intersection point is encoded to represent the secret. Each secret share is constructed as a hyperplane in a vector space over a finite field. Thus, the secret can be reconstructed only if at least t hyperplanes are available. To rebuild the secret, the point of intersection of the hyperplanes are to be found out by solving the hyperplane equations.

Apart from Shamir and Blakley schemes, there are two major threshold cryptography schemes viz., Mignotte's [6] and Asmuth-Bloom's schemes [7], using the Chinese Remainder Theorem (CRT). With Chinese Remainder Theorem, a number can be regained from a set of congruence equations. The congruence equations are relations which express the number modulo many relatively prime integers. This notion is used to construct a secret from a set of shares which are congruence equations for the secret. In order to create a (t, n) threshold scheme, n congruence equations are constructed using integers $m_1, m_2, m_3 \ldots m_n$ that are relatively prime to the secret, s. The integers are chosen in such a way that $s < \prod_{i=1}^{n} m_i$. At the same time, s should be greater than the product of any $(t-1)$ combination of $m_1, m_2, m_3 \ldots m_n$. This essentially constructs a (t, n) threshold scheme where the secret s can be reconstructed only when t or more congruence equations are known. Mignotte's and Asmuth-Bloom's techniques use this idea in the core but vary in the way they choose the relatively prime integers. While Mignotte's uses a special sequence of integers called the Mignotte Sequence, Asmuth-Bloom's also uses a sequence holding a few unique properties.

The work by Lein Harn [8], proposes a new type of authentication - group authentication denoted as $GAS(t, m, n)$ where t is the threshold for the scheme, m is the number of group members and n the total number of users, to authenticate all users belonging to the same group at once. The basic idea behind the work is as follows. All members of a group use their pre-shared secret share to reconstruct a secret value, which should match with the stored value. If the secrets do not match, the scheme indicates that there are non-members in the group. The scheme relies on Shamir's Secret Sharing scheme to generate secret shares and reconstruct the secret.

III. Shamir's Secret Sharing Scheme

Shamir's Secret Sharing scheme [4] can be used to split an integer, s to n shares. The secret integer, s can be reconstructed successfully if a threshold number, t of shares are available. The algorithm to split the integer is known as the Dealing algorithm and the Reconstruction algorithm is used to reconstruct the secret. The Dealing algorithm is given in Algorithm 1 and Reconstruction algorithm is as in Algorithm 2.

Algorithm 1 Shamir's Secret Sharing Scheme: Dealing Algorithm

Input: Number of shares n, Threshold value t, a prime number q and Secret integer $s \in \mathbb{Z}_q$
Output: A list of n shares, $\zeta = \{s_i\}$ *where* $1 \leq i \leq n$
  Randomly choose $t-1$ random integers $a_1, a_2, a_3 \ldots a_{t-1}$
  $a_0 \leftarrow s$
  Build the polynomial
  $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$
  for $i \leftarrow 1$ *to* n **do**
    $s_i \leftarrow (i, f(i) \mod q)$
  end for
  return ζ

Algorithm 2 Shamir's Secret Sharing Scheme: Reconstruction Algorithm

Input: A list of shares, $S = (x_i, y_i) \forall i > 0$
Output: The reconstructed value, s'
  for *each* $j \mid (x_j, y_j) \in S$ **do**
    $\ell_{j,0,S} \leftarrow \prod_{\substack{(x_m, y_m) \in S \\ m \neq j}} \frac{-x_m}{x_j - x_m} (mod \ q)$
  end for
  $s' = \sum_{(x_j, y_j) \in S} y_j \ell_{j,0,S} (mod \ q)$
  return s'

IV. Multi-factor Authentication Using Threshold Cryptography

The idea behind our proposed technique is to use Shamir's Secret Sharing scheme to generate the One Time Passwords

(OTP) by splitting a random secret and to reconstruct the secret back when a threshold number of OTPs are fed back. The process involves a secure authentication server that performs the authentication process using Shamir's Secret Sharing scheme. The process involves the following phases.

A. Registration

As with any authentication system, the users should register in the system with their credentials. The credentials should include a username to uniquely identify the user. The user should also provide the mobile phone number, email address, and other identifiers through which the OTPs are to be sent. If the user intends to use his/her own secret (PIN or password) as one factor of authentication, the secret should also be fed to the system in this phase.

B. Initialization

The initialization of the process starts when a user tries to authenticate to a system. The user may be either trying to login to an application or authorize a banking transaction which uses multi-factor authentication. Once the user indicates to get authenticated, the authentication server generates a cryptographically secure random number s, and stores it securely in its database.

C. One Time Password Generation

In this phase, the authentication server uses Shamir's Dealing Algorithm to split the random number to secret shares, following a $t(t, n)$ threshold scheme, where n is the number of authentication factors used and t is the threshold value. The server then sends each share to the user through the different channels viz., the smartphone application, SMS, and email. If the user has specified to use a secret (PIN or password), the shares are generated using a modified version of Shamir's dealing algorithm as given in Algorithm 3. The algorithm splits the random number to n secret shares, out of which one share is set as the user's secret.

Algorithm 3 Modified Dealing Algorithm

Input: Number of shares n, Threshold value t, a prime number q and Secret integer $s \in \mathbb{Z}_q$, a predefined share S

Output: A list of n shares, $\zeta = \{s_i\}$ *where* $1 \leq i \leq n$ and $s_1 = S$

Randomly choose $t-2$ random integers $a_2, a_3 \ldots a_{t-1}$

$a_0 \leftarrow s$

$a_1 \leftarrow S - (a_0 + a_2 + \ldots + a_{t-1})$

Build the polynomial

$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$

for $i \leftarrow 1$ *to* n **do**

  $s_i \leftarrow (i, f(i) \mod q)$

end for

return ζ

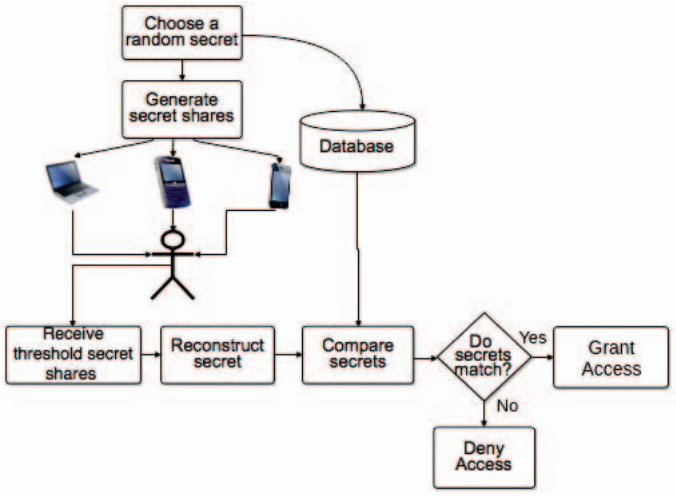

Fig. 1. Illustration of multi-factor authentication

D. Authentication

Once the user receives the threshold number of OTPs, he/she needs to send them back to the server. The server waits until it receives the threshold number of OTPs from the user. Upon receiving, the server tries to reconstruct the secret using the Reconstruction Algorithm of Shamir's scheme. The server then compares the reconstructed secret with the original secret random number stored in the database. The user gets authenticated if the comparison yields a match. The server denies authentication if the secret random number does not match with the reconstructed value.

The process is illustrated in Fig. 1.

V. Implementation of the Proposed Scheme

An Android application, for authorizing payments at *Point-of-Sale* (PoS) terminals, was built to simulate the proposed multi-factor authentication scheme. The application communicates with an authentication server to receive the OTP and send user fed OTP back to the server. The final status of authentication is also sent back to the application. The authentication server runs a $t(3, 2)$ scheme so as to authenticate a user when it receives back either the OTP delivered by SMS or email, apart from the OTP sent to the application. The authentication process is illustrated in Fig. 2.

The simulation process is detailed below.

1) The shopkeeper or waiter swipes the debit or credit card.
2) The server then generates a pseudo random number, n and stores it for the final comparison.
3) The random number n, is then split into three shares s_1, s_2, s_3 using Shamir's Secret Sharing scheme setting the threshold to 1. The secret random number is split in such a way that one out of the three shares will be the user's secret ATM PIN (s_1).
4) The server then sends one share (s_2) to the user's registered mobile number as an SMS and the third share (s_3) to the mobile application.

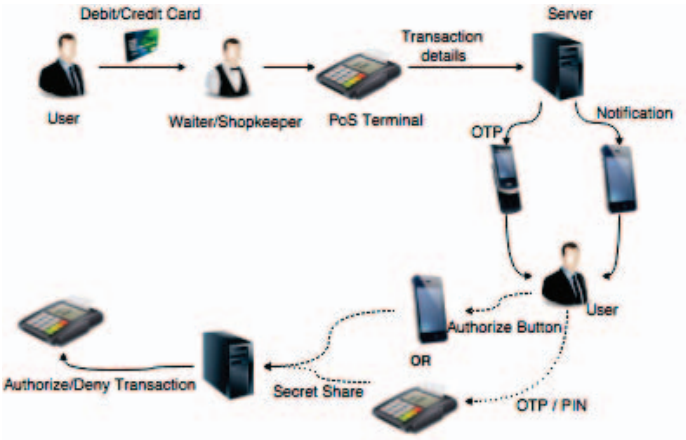

Fig. 2. Authentication at a *Point-of-Sale* terminal using the proposed technique

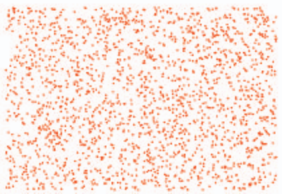

Fig. 3. Randomness of OTPs. To plot a six digit OTP, the first three digits are plotted on the $x-axis$ and the last three digits are plotted on $y-axis$

TABLE I
COMPARISON OF EXISTING AND PROPOSED n-FACTOR MFA SCHEMES

Parameter	Existing MFA	Proposed MFA
Number of comparisons	n	1
Number of database reads/writes	n	1
Storage required	n	1

5) The user to authenticate himself can perform any of the three actions:
   a) Key in the ATM PIN, if he feels there is no security issue
   b) Key in or ask the waiter/shopkeeper to key in the OTP received as SMS
   c) Launch the mobile application in the smartphone and click the displayed 'Authorize Transaction' option. The mobile application effectively sends back the received share to the server.
6) The server then waits till it receives, at least, one share from any of the channels.
7) Once the server receives a share from any of the devices it reconstructs the secret random number n' using the reconstruction algorithm given in Algorithm 2.
8) The server then compares n and n' and authorizes the transaction if n and n' are the same. The server then sends the acknowledgement to the PoS device.

VI. ANALYSIS

A. Security

As pointed out in Section I, current multi-factor authentication schemes possess security issues when using user's long-term secret in public places such as PoS terminals and ATM booths. Asking users to expose their secret in public places is a fundamental mistake and is practised in millions of transactions worldwide every day. The proposed scheme effectively eliminates the security risk involved in using secrets in public places for authentication. The scheme, using a $t(1,n)$ Threshold Cryptography, can be used to employ an authentication system where a user may choose to enter anyone of the available n secret shares to complete a transaction. The users may choose to enter the long-term secret whenever they feel that the secret can be entered without letting anyone among the crowd to shoulder surf the PoS or ATM device's keypad. In case of shoulder surfing or a similar security vulnerability the user may choose to enter the OTP instead of the secret, thus preventing the exposure of user's secret to the public. Since the OTP is a random number and is very unlikely to be reused in the near future, its exposure to the public is safe. The randomness of the OTP can be ensured by employing cryptographically safe pseudo-random number generators [9]. In the simulation presented in Section V, random numbers are generated from the $/dev/urandom$ file available in Linux machines. Fig. 3 plots 2000 six digit OTPs generated by the proposed scheme on (x,y) plane. This method is adapted from [10]. The first three digits of the OTPs are plotted on the $x-axis$ and the last three digits are plotted on the $y-axis$. It can be seen that the points on the graph are uniformly distributed, which indicates that the OTPs hold fair randomness and a replay attack by an adversary will be unfruitful using OTPs stolen through shoulder surfing.

B. Storage efficiency

In multi-factor authentication, each factor is associated with a secret. The secret can be either the user's password or PIN or an OTP. All secrets used during an authentication session should be stored securely till the process completes. On the other hand, the proposed scheme requires only one secret to be stored in the database during the lifetime of the authentication session. For an n-factor multi-factor authentication scheme, the proposed scheme thus reduces the number of secrets to be stored to one against n secrets to be stored in the case of the existing scheme.

C. Performance

In existing multi-factor authentication schemes, all OTPs used are compared at the end of the process to authenticate the user. Processing power is also consumed by the data storage and retrieval processes. An n-factor scheme thus involves n comparisons, n database writes and n database reads. In the proposed scheme, both the number of comparisons and database I/O operations are reduced to one. This is because only one secret (random number) is stored for later comparison. Table I summarizes the comparison results.

D. Flexibility

Introducing a new channel to share OTPs require an extra run of the OTP generator in existing multi-factor authentication systems. The proposed scheme is flexible and requires only to change the *number of shares* (n) parameter of the authentication system to add an extra channel. The scheme does not require the OTP generator (Shamir's Dealing Algorithm in our scheme) to run multiple times for the generation of different OTPs. Also, users may feed back the OTPs in any order as per their convenience. Many of the current multi-factor authentication systems require users to key-in back their OTPs in the order of the authentication programs used.

VII. Conclusion

Multi-Factor Authentication is a way of authenticating users using multiple layers of authentication programs. It is considered to be a secure way of authentication that can effectively prevent online identity theft. The work proposed a new scheme to improve the technique on various parameters using threshold cryptography. The improved authentication scheme is successfully simulated and evaluated using an Android application. The scheme improves the security, convenience, flexibility, storage efficiency and performance of multi-factor authentication. The technique can be improved to include stronger authentication methods like biometrics. This would make the technique suitable for high-security systems. The proposed scheme, thus, can improve the existing solutions to security, privacy and authentication problems in various areas including mobile, cloud and internet computing.

References

[1] K. Abhishek *et al.*, "A comprehensive study on multifactor authentication schemes," in *Advances in Computing and Information Technology: Proceedings of the Second International Conference on Advances in Computing and Information Technology (ACITY) July 13-15, 2012, Chennai, India - Volume 2*, N. Meghanathan *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 561–568.

[2] J. Pieprzyk *et al.*, "Secret sharing," in *Fundamentals of Computer Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 327–351.

[3] H. Delfs and H. Knebl, "Cryptographic protocols," in *Introduction to Cryptography: Principles and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 81–133.

[4] A. Shamir, "How to Share a Secret." *Communications of the ACM 22.11*, vol. 22, no. 11, pp. 612–613, 1979.

[5] G. R. Blakley *et al.*, "Safeguarding cryptographic keys," in *Proceedings of the national computer conference*, vol. 48, 1979, pp. 313–317.

[6] M. Mignotte, "How to share a secret," in *Cryptography*, ser. Lecture Notes in Computer Science, T. Beth, Ed. Springer Berlin Heidelberg, 1983, vol. 149, pp. 371–375.

[7] C. Asmuth and J. Bloom, "A modular approach to key safeguarding," *Information Theory, IEEE Transactions on*, vol. 29, no. 2, pp. 208–210, Mar. 1983.

[8] L. Harn, "Group authentication," *IEEE Trans. Comput.*, vol. 62, pp. 1893–1898, Sep. 2013.

[9] S. Crocker and J. I. Schiller, "Randomness Requirements for Security," RFC 4086, Mar. 2013.

[10] A. Dmitrienko *et al.*, "Security analysis of mobile two-factor authentication schemes," *Intel Technology Journal*, vol. 18, no. 4, pp. 138–161, 2014.