Secret P-box based Mutual Authentication Mechanism for Cloud Computing

Mandeeep Kaur[1]† and Dr. Prachi Garg[2]††,

†Research Scholar, CSED, Maharishi Markandeshwar University(Deemed to be University), Ambala & Assistant Prof., PIET, Samalkha, Panipat
††Associate Professor, CSED, Maharishi Markandeshwar University(Deemed to be University), Ambala

Abstract

Cloud computing is an emerging business model popularized during the last few years by the IT industry. Providing Everything as a Service has shifted many organizations to choose cloud-based services. However, some companies still fear shifting their data to the cloud due to issues related to the security and privacy. The paper suggests a novel mutual authentication mechanism using Secret P-box based Mutual Authentication Mechanism (SPMAM) on the criticality of information. It uses a particular passcodes from one of the secret P-box to act as challenge to one party. The response is another passcode from other P-box. The mechanism is designed in a way that the response given by a party to a challenge is itself a new challenge for the other party. Access to data is provided after ensuring certain number of correct challenge-responses. The complexity can be dynamically updated on basis of criticality of the information and trust factor between the two parties. The communication is encrypted and time-stamped to avoid interceptions and reuse. Overall, it is good authentication mechanism without the use of expensive devices and participation of a trusted third party.

Keywords:

Access control, Mutual authentication, Cloud computing, Data Security,

1. Introduction

Cloud computing is emerging as a technology that has changed the use of hardware, software and services by organizations using parallel distributed computing systems comprising inter-connected virtualized computers [1]. It may be viewed as a stack of Applications, Platforms and Infrastructure provided “as service” by the service providers. It runs over virtual machines and applications that work on any operating system and provides services using the internet [2,3] on a pay-per-use cost model. It improves cost overheads borne by the organization on maintenance and upgradation of hardware/software.

The clients can access the cloud services after passing the authentication test based on credentials recorded during user signup. Authentication is a mechanism that verifies the validity of the claimed identity of an individual based on something an individual knows, possesses, is or does. It is an important step in securing information [4,5]. In a cloud computing environment, the authentication of valid users is much more critical it opens the access to entire information set of the organization. Providing secure access to the information placed over the cloud is a big issue (see Fig. 1). The security of data placed on cloud servers is one of the biggest challenges in adopting cloud services according to RightScale (now Flexera) studies from 2015 to 2022 [6,7,8,9,10,11,12,13]. Still, organizations' IT spending is slowly shifting from traditional offerings to cloud services. It is predicted that the cloud market will be $216 billion in 2020. New non-conventional methods of secure login and authentication on public cloud servers are required [14,15] rather than using behavioral metrics or establishing expensive private clouds for collaborative work as a solution to security problems.

This paper is organized as follows. The next section describes the methodology we followed int his paper. Section 3 discusses the companies’ survey results. In section 4, we discuss the results of the students’ survey. Section 5 summarizes the results and gives some recommendations. Finally, in section 6, we give some concluding remarks.

2. Related Work

Mathematical foundations have always been catalysts in the design of authentication, security and encoding techniques. The researchers are creating authentication techniques using passwords. Identity can be established using OTP for authentication of the digital identity of the user [16]. A two-factor hashed OTP-based authentication [17] using MD5 has also been advocated. N-screen-based consolidated authentication to access various devices [18] look good in slashing time overheads. Single sign-on for

Manuscript received December 5, 2024
Manuscript revised December 20, 2024
https://doi.org/10.22937/IJCSNS.2024.24.12.7

reliable access to cloud software-as-a-service [19] uses Secure Socket Layer and Advanced Encryption Standard cryptographic algorithm for increasing security. The authentication and Leak Prediction Model (ALP) [20] uses redacted trees. Identity-based authentication schemes use public key encrypted certificates [21] and a flexible combination of OTP with TLS standard [22] can be used for better security during authentication. Newer methods use a graphical sequence of images given to a user to create a sequence [23] for authentication. S. Furnell et. al [24] used 3D graphical passwords with dynamic challenges in 3D structures to increase the complexity of guessing the passwords. The dynamics of the method could be improved by adding more graphical operations to it. Y. Yang et. al [25] presented authentication based on doodling a particular pattern in a square grid of the screen. Older users do not like these methods much as these were knowledge-based, having a dependency on the touch screen and its size Z. Zhao et. al [26] and D.

Nyang et. al [27] presented gesture-based access to a device or service captured with a camera. It is safer when used as a re-authentication of a user in addition to password-based access. These may pose critical issues while used as a primary authentication mechanism due to the ease of copying the gestures once noticed by other users. Use of keystroke dynamics [28] for authentication proposed by P. S. Teh et. al [29] as biometric information used statistics of typing profiles to identify a user. Gestural and keystroke dynamics were combined to provide an improved method of continuous authentication using an AI-based machine learning algorithm by J. Wu et. al [30]. Behavioral activities captured by sensors are used to relate humans with their behavioral activities suggested by M. N. Aman et. al [31] and Liang Y. et. al [32]. The self-driven automated procedure is hardware dependent and could deviate from its normal working in certain scenarios. It also leads to a breach of privacy laws. Bansal, G. et. al. [33] proposed lightweight Secure User Key-Exchange Authentication (SUKA) for a two-step mutual authentication of vehicles in an IoT environment. This is based on tamper-proof chips installed in vehicles. Shashidhara, R. et. al [34] designed a lightweight mutual authentication system that is global roaming efficient and robust. It uses old security algorithms that consume fewer resources but might be compromised. R. Ferrero et. al [35] guided to use gait-based recognition of users by inputs from an accelerometer. It is Smartphone dependent and may not uniquely authenticate users if there are more users. Fantana, A. L. et. al [36] provided a design for a movement-based biometric authentication using smartphone movement records. However, the performance of this method varies with different brands of smartphone models.

2.1 Issues

There is no single authentication solution that may be applied convincingly and appear foolproof on paper [37] to a cloud environment comprising of machines/devices as users like in an IoT environment. Techniques like passwords, security certificates, virtual private networks and cryptographic algorithms do not appear sufficient due to the absence of a key-in facility for devices. The use of biometric identification, verification and templates of voice, face or retina suffers from internal bias [38] and requires biometric devices and storage for high-resolution biometric images [39,40]. Problems associated with changes in biological metrics like shape, color, voice pattern and variations in the environmental conditions cause these methods to fail at times. Behavio-metric authentication systems are application and user behavior dependent that may be modified [41]. It is difficult to standardize this method as users can vary their behavior due to their mood and health. OTP-based methods tend to increase overall authentication time while N-screen-based consolidated authentication [18] increase the interest of intruders as one crack enable access to multiple devices. Single sign-on has considerable overhead due to the use of heavy cryptographic techniques. The Authentication and Leak Prediction Model (ALP) [20] can be misused. Identity-based authentication schemes are device-dependent and can be copied/cloned and pose problems in case of device theft. A flexible authentication solution using OTP and TLS standards was also suggested to work for different security settings similar to using digital certificates.

2.2 Problem Formulation and Objective

The paper aims to develop a user authentication algorithm based on a dynamic challenge-response approach that validates users without expensive biometric devices and a Trusted Third Party (TTP). Moreover, depending on the criticality of the information to be accessed, the complexity of the algorithm may be increased or decreased. It employs encrypted communication for less susceptibility to security attacks.

A Secret P-box-based Mutual Authentication Mechanism (SPMAM) has been proposed that works for all clients' kind: users or devices in accessing cloud services. It uses a randomly chosen sequence of challenges-responses between two parties under communication to authenticate each other (see Fig. 2) without involving trusted third party. The complexity of algorithm is dynamically controlled based on trust factor. It applies encrypted communication to make it less susceptible to attacks. Section 3 provides details of the proposed design. Section 4 elaborates experimental setup used to evaluate its performance evaluation and results that are provided in Section 5.

3. Proposed SPMAM Framework

SPMAM uses dynamically generated challenges and responses for the mutual authentication of two parties under communication. The client organization registers for cloud service. The administrator decides the number of classification tiers, number of iterations, sizes of Passcode boxes, and encryption-decryption algorithm to be used in the authentication.

3.1 Parameter Table & Data Classification

The parameters and the notations used have been explained in Table 1.

Table 1. Parameters and Notations Used

Parameter Notation	Description
I	Number of Iterative rounds in the proposed framework
c	Tier Level (0-4)
u	Number of users in a tier group
T	Trust Factor (real number between 0 and 1)
A	Total number of authentication attempts made by the user
B	Number of unsuccessful authentication attempts
S	Security multiplier (positive integer)
s_p	Size of P-box
F()	Passcode extractor function
$EK_i()$	Encryption Algorithm
$DK_i()$	Decryption Algorithm
AM	Authentication Manager program
Passcode	A sequence of characters from a set of Alphabets
Rn	A random number generated by the user with value between 1 to s_p for the server
Ru	A random number generated by the server with value between 1 to s_p for the user
Tn	Timestamp for server
Tu	Timestamp for user
IsUserBox	Boolean value. 1 means value to be extracted from user's P-box and 0 means value extracted from server's P-box
p_{uj}	Passcode from user's P-box during j-th iteration
P_{nj}	Passcode from server's P-box during j-th iteration

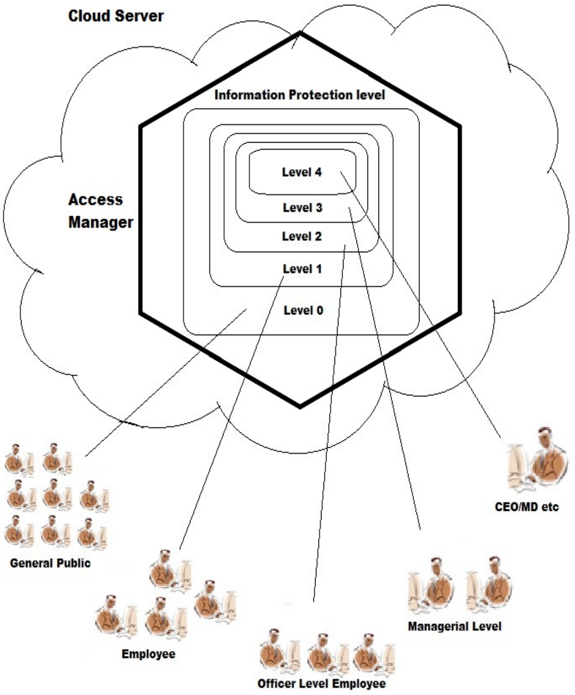

Figure 3. Suggested Accessibility Framework

Access control is divided into five tiers: Level 4, Level 3, Level 2, Level 1 and Level 0 with decreasing level security. Members of the higher tiers can access data from all lower tiers upon successful authentication. The number of users in the higher tier is less than the lower tier (see Figure 3).

Classification of data is done into five categories: Top secret, secret, confidential, restricted and public [42] based on its impact and criticality (refer Table 2). The cloud server and the client organization choose some agreed-upon symmetric or asymmetric encryption algorithm, say $EK_i()$ to encrypt the group passcodes used for authentication [43,44].

Table 2. Data Classification Tiers

Data Classification	**Impact of unauthorized Access**	**Tier Level (c)**
Top Secret	Would cause "exceptionally grave damage" to the organization/nation.	4 (Highest level of criticality)
Secret	Would cause "serious damage" to organizational/national security if it were publicly available	3 (Third level of criticality)
Confidential	Would cause "damage" or be "prejudicial" to Top organizational/national security	2 (Second level of criticality)
Restricted	Would cause "undesirable effects" if publicly available.	1 (First level of criticality)
Public	No harm and can be made publicly available.	0 (lowest criticality)

Each user is allocated some tier level by the administrator. The number of iterative rounds (I) of each tier is given by equation (1).

$$I = S*2c* \log_2 u / T \qquad (1)$$

Initially, the value of trust factor T is set to 0.1 or 0.2. It is changed after a specific number of authentication attempts as per equation (2)

$$T = \text{Min.}\{1, T(1+(A-0.33*B)/A)\} \qquad (2)$$

Equation (2) guarantees to increase the value of trust for three percent or fewer authentication failures and decrease it for a higher percentage of failures. The inverse relation between T and I increases number of iterations when T decreases and vice-versa.

3.2 Components and Functionalities

SPMAM is implemented using a secret square P-box and APIs consisting of function F(), encryption algorithm $EK_i()$ and Authentication Manager program. The clients are provided with these components for mutual authentication. A brief description of the components and the functionalities follows.

3.2.1 Secret Square Passcode–box (P-box)

A secret square P-box contains passcodes in rows and columns. The size of the P-box is determined based on the number of users and the Trust factor as described in Table 3.

Table 3. Size of P-box for various tiers of information

Tier Number (i)	**No. of persons in Tier**	**Dimension of square P-box (To Integer value)**
0	n0	Not required
1	n1	$\log_2$(n1)*i^i/(T)
2	n2	$\log_2$ (n2)*i^i/(T)
3	n3	$\log_2$ (n3)*i^i/(T)
4	n4	$\log_2$ (n4)*i^i/(T)

Users provide a set of characters that may be used to create passcodes of some fixed length. Such passcodes are filled in their respective square P-boxes. These P-boxes are stored with the cloud server. The Cloud server also provides a same-sized P-box filled with similar passcodes to the users and sends it to the client user. Figure 4 shows an example 3x3 P-box containing 3-length passcodes consisting of digits and characters.

3.2.2 Function

Function F() is part of the API used to interact with the P-box using three arguments. It helps extract a passcode from a particular row number and column number of the P-box. For example, if the user and the server have opted for the P-boxes as mentioned in Fig. 4, the function F(Server,2,3) extracts the value Zw5 at row 2, column 3 of the server's P-box. In the same way F(User, 3,1) extracts Za4 from the user's P-box.

3.2.3 Encryption Algorithm

The client and the cloud server choose some encryption algorithm $EK_i()$ to be used for encrypting information passed over the internet during authentication. The corresponding decryption algorithm $DK_i()$ is used to decrypt and extract the plaintext.

User P-box

As3	Np7	Qw2
Ik9	Lh1	Ve8
Za4	Bf5	Uq6

Server P-box

Xp2	Kl9	Ur6
Gj0	Mr7	Zw5
Jp1	Dv3	Sy4

Figure 4. P-box with example passcodes

3.2.4 Authentication Manager

The clients are provided with an AM program that is used at the client end and the server end for mutual authentication. It uses P-boxes, F(), EKi() and DKi(). It generates random numbers R_n and R_u between the range $(1,s_p)$ for use in the authentication process.

3.2.5 Passcode Generation and Storage

Individual users must register with the cloud server. The user must go through the following steps to register.

Step 1: The user chooses the username, organization and tier level he belongs to.

Step 2: The user decides the length of passcodes, its set of alphabets (contains valid symbols used to generate passcode) and the format for making passcodes. It facilitates country or language-specific customized data sets to be used in passcodes.

Step 2: A random generator may be used to generate passcodes that are filled in the P-box by the user. This square P-box is stored with the cloud server and the user.

Step 3: In the same manner, the cloud server fills a P-box of the same size with its passcodes. It is also stored at the user end and the cloud server.

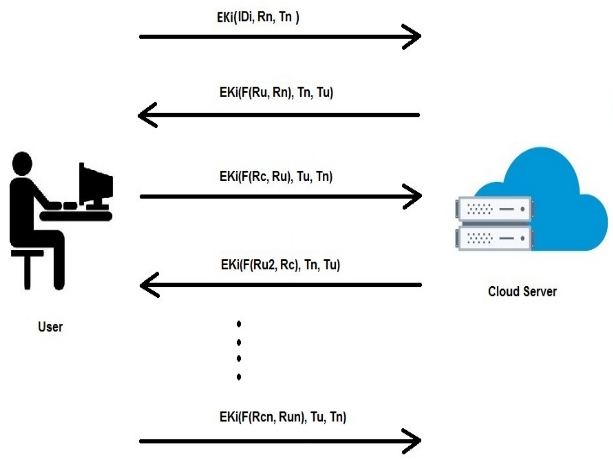

Figure 5. Mutual Authentication Process

3.3 Authentication Process

The mutual authentication process relies on multiple challenge-response iterations between parties under communication. It uses a particular passcode from one of the P-box to act as a challenge to one party in the communication. The response to this is another passcode from the other P-box which is related to the previous passcode in some way. This response passcode also acts as a challenge to the other party. The process ensures that the interceptor is not able to predict the passcode and its exact position in the P-box as the entire communication is encrypted using EKi. The step-wise algorithm is explained further and the overall process is shown in the flowchart in Fig. 5.

SPMAM *Algorithm*

The P-box structures are loaded by the Authentication Manager program and the following steps are carried out.

Begin

Step 1: The user enters the username and selects the classified tier to be accessed. The server checks the username and tier combination in registered candidates.

Step 2: If the combination exists, direct the user to the authentication interface and load the number of iterations (I) for that tier else goto End.

Step 3: Set iteration number J:=1.

Step 4: The user sends the encrypted information containing his ID, a randomly generated value R_n and timestamp T_n to the server.

Step 5: The server generates a random number R_u. Set IsUserBox=1. It then evaluates the value at row R_n and column R_u of the user's P-box using the function p_{uj}=F(IsUserBox, R_n, R_u). It generates timestamp T_u.

Step 6: It uses $p_{uj}||T_n||T_u$ to concatenate the three values, encrypt it using EKi($p_{uj}||T_n||T_u$) and sends it to the user.

Step 7: The user decrypts the value received from the server using $DK_i()$. It extracts T_n. If the value does not match the sent value, the authentication process is stopped by the transfer of control to Step 14.

Step 8: The timestamp T_u is extracted. If (T_u>T_n) then passcode value F(IsUserSide,R_n,R_u) is extracted. If the received passcode value p_{uj} matches with some code in column number R_u of the row R_n, the number sent by server is traced.

Step 9: The user generates another random number R_{n2}. Set IsUserBox=0. It then evaluates the value at row R_{n2} and column R_u of the server's P-box using function p_{nj}=F(IsUserBox, R_{n2}, R_u). It generates timestamp T_{n2}.

Step 10: It uses $p_{nj}||T_{n2}||T_u$ to concatenate the three values, encrypt it using EK_i($p_{nj}||T_{n2}||T_u$) and sends it to the server.

Step 11: The server decrypts the value received from the user using $DK_i()$. It extracts T_u. If the value does not match the sent value, the authentication process is stopped by transfer of control to Step 14.

Step 12: The timestamp T_{n2} is extracted. If (T_{n2}>T_u) then passcode value F(IsUserSide,R_{n2},R_u) is extracted. If the received passcode value p_{nj} matches with some code in row number R_{n2} of the column R_u, the number sent by the user is traced.

Step 13: Increment the iteration number J by 1 and while(J<I) goto Step 5.

Step 14: If(J==I) then "Grant access to the tier" else Issue warning of "Mutual Authentication Failed"

End

4. Experimental Setup

Simulation of the proposed framework is done using Turbo C program on a 64-bit machine with Intel(R) CPU Core i3 M370 @ 2.40 GHz processor with 3 GB RAM. An environment with different tiers of information and other components of the suggested framework is created. Initial registration of dummy users is performed with their categorization in tiers and some number of users in the tier is set. The program recorded the time taken for authentication of different categories with a random number of users for different tiers, different sizes of square P-boxes and various lengths of passcodes. It also tracked the variation in trust factor value for 50 iterations of 100 login authentication attempts each by assuming some percentage of authentication failures. The experiment is repeated to record the average performance of the algorithm.

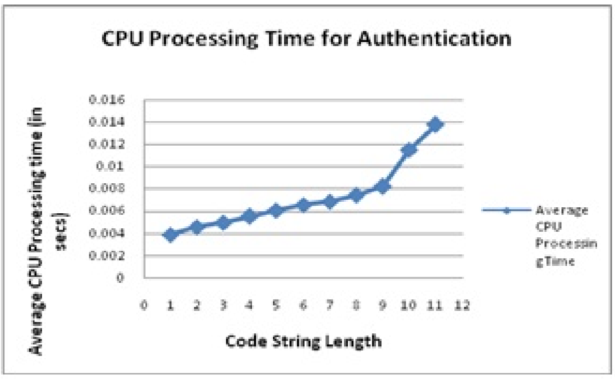

Figure 6. Processing time for Authentication

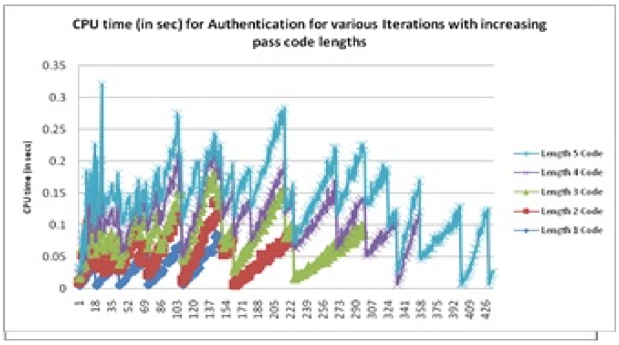

Figure 7. CPU time for authentication in various iterations with increasing passcode length

5. Performance Evaluation

The performance of SPMAM is evaluated based on CPU processing time, the time required for a brute force attack and variation of trust factor to check its performance.

5.1 CPU Processing Time

The average CPU time taken by the SPMAM algorithm increases almost linearly with the increase in length of the passcode string upto length 8 (see Fig. 6). An increase in the average CPU processing time was noticed for passcode string lengths of 9 or more. This is attributed to the increase in the size of characters (digits and alphabets) beyond its 64-bit in unit time. The CPU time for authentication for iterations carried with different size of passcode lengths is shown in Fig. 7. The average time taken for authentication increases with the increase in the length of passcodes or the increased number of iterations as expected.

5.2 Brute-force Attack

The complexity of carrying a brute-force attack on the proposed algorithm is calculated for different passcode lengths. Table 3 shows the number of key space options to explore as per the size of the passcodes comprising some number of characters and digits. It is observed that longer-length passcodes take more time to crack on an average desktop with almost 17 billion tries in an hour for key-space searches [45]. It is observed that cracking passcodes of length 10 or more in the P-boxes will be sufficiently complex. The estimated hours required to crack a passcode of length 7 or more using a single or distributed computing environment comprising at most 500 machines is safe for almost 342 days for I=1. For I >1, the time complexity for correctly guessing passcodes multiple times will increase further and strengthen the security

5.3 Growth of Trust Factor

The trust factor dynamically controls the iterations of authentication. It is observed (see Fig. 8 and 9) that starting with some low initial trust value if the number of unsuccessful attempts can be kept below 3%, the trust value tends to increase over time. A higher proportion of authentication failures may also lead to the reduction of the trust factor to almost zero. It shows that the value of trust reach 1 (highest) value in few runs and vary with failures to increase or decrease the iterations used in the authentication process.

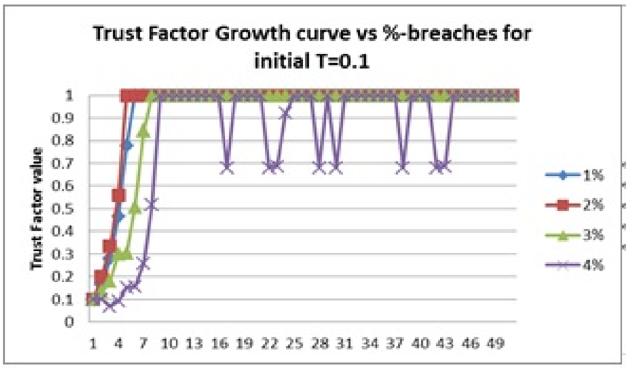

Figure 8. Trust factor growth chart for SPMAM with iterations

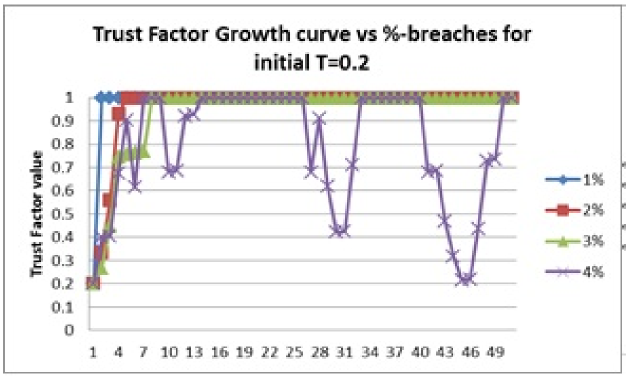

Figure 9. Trust factor growth chart for SPMAM with iterations

5.4 Strengths

The passcode exchange is encrypted before sending which avoids man-in-the-middle attack. Moreover, the use of timestamps nullifies the replay attack. Even if someone identifies a passcode, its position in the P-box cannot be predicted accurately as either row or column value will be known with surety. The number of iterations can be increased or decreased to increase the complexity of computation for a brute-force attack. The algorithm provides tier-based dynamic number of iterations using random passcodes based on the trust factor. It does mutual authentication without the participation of a trusted third party. The algorithm is backward compatible with password-based authentication that may use single iteration of the proposed algorithm. In a nutshell, the new technique has high level of complexity for intruders and makes the prediction of dynamic passcode during a series of alternate challenge-response methods.

Table 4. Processing time in hours to undertake a brute-force attack on single Passcode of different lengths

Upper Case Letters	6	6	6	7	8	9	10
Numbers	1	2	3	1	1	1	1
password length in Characters	7	8	9	8	9	10	11
Number of Combinations	3 billion	30 billion	308 billion	80 billion	2 trillion	54 trillion	1 quadrillion
Estimated hours to crack using a single machine	0.08990	0.899063	8.99063	2.337564	60.7766	1580.19	41085.0
No. of days to crack using single machine →	0.0037	0.037460	0.37460	0.097398	2.53236	65.8413	1711.87
Estimated hours to crack using Distributed level with number of machines ↓							
10	0.00899	0.089906	0.89906	23.37564	607.766	15801.9	410850.0
50	0.00179	0.017981	0.17981	4.675128	121.55	3160.38	82170.0
100	0.00089	0.008990	0.08990	2.337564	60.7766	1580.193	41085.0
250	0.00035	0.003596	0.03596	0.935025	24.3106	632.077	16434.0
500	0.00017	0.001798	0.01798	0.467512	12.155	316.038	8217.00

Table 5. Comparison of various methods of Authentication with proposed SPMAM

Method- > Feature | V	**Proposed SPMAM Algorithm**	**Authentica-tion score method**	**OTP based**	**Hashed MD5**	**Two factor method**	**Session key method**	**IP Sec Mutual Authentication**	**Biometric methods**	**Identity based Public Key Certificates**	**User Behaviour model**
static Password based	x	x	x	x	x	√	X	x	x	x
Application dependent	x	√	x	x	√	√	X	x	√	√
Device dependent	x	√	√	x	√	√	X	x	√	x
Undue overhead	x	x	x	√	x	X	X	x	x	√
Biometric anomalies	x	x	x	x	x	X	X	√	x	√
Issue Certificate	x	x	x	x	x	x	√	x	√	x
Requires same or trusted domains	x	x	x	√	x	√	√	x	√	√

Requires Trusted Third Party	x	√	x	x	x	x	X	x	√	√
Uses Database repository	√	x	x	√	x	x	X	√	√	√
Can be grown/ shrink dynamically	√	x	x	x	x	x	X	x	x	x
Uses cryptographic method	√	x	x	√	x	x	X	x	√	x

6. Comparison with related work

A comparison of the proposed SPMAM is shown in Table 5 with other existing popular ten authentication methods. It shows eleven parameters on which the comparison has been done. It helps to decide which one would be useful in a particular kind of situation

7. Conclusion

Security of data stored on cloud servers has always been a significant concern. The security issue can be substantially resolved by access to data only by legitimate users based on a strong authentication mechanism. The time invested in such a process shall increase the overall safety of data on the cloud. The solution suggested in this work uses passcodes of a specified length from a pool of codes to mutually authenticate valid users and servers. The passcodes act as a challenge and response at the same time for verification of parties under communication. The complexity of the algorithm can be increased or decreased dynamically based on the trust factor, number of iterations, passcode length and set of allowed alphabets. It is backward compatible to support low-efficient machines by taking a single iteration. The use of timestamps helps in revoking replay attacks, key size makes it difficult to apply brute-force attacks. The use of encryption during communication helps in overcoming man-in-the-middle attacks. In a nutshell, SPMAM appears to be a good mutual authentication algorithm that does not require a trusted third party to execute and provide appropriate security using according to the classification of information, trust factor and encrypted communication.

Acknowledgements

We acknowledge the labs and library section of Panipat Institue of Engineering and Technology, Samalkha, Panipat and MMU, Mullana for the access of the resources.

References

[1] NIST definition on cloud computing accessed from nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[2] James Broberg, Andrzej Goscinski, Rajkumar Buyya, Cloud Computing: Principles and Paradigms, Wiley, Part VI, 2013.

[3] Judith Hurwitz, Robin Bloor, and Marcia Kaufman, Cloud Computing For Dummies, Wiley Publishing Inc.,Indiana, 2010.

[4] Stallings W., Cryptography and Network Security, Principles and Practices, Fourth Edition, Pearson Education.

[5] Mark Stamp, Information Security Principles and Practice, Wiley India Pvt. Ltd., pp. 215-220, 2006.

[6] RightScale 2015 State of the Cloud Report, p. 20.

[7] RightScale 2016 State of the Cloud Report, p. 19.

[8] RightScale 2017 State of the Cloud Report, p. 16.

[9] RightScale 2018 State of the Cloud Report, p. 21

[10] RightScale 2019 State of the Cloud Report, p. 21.

[11] Flexera 2020 State of the Cloud Report, p. 37

[12] Flexera 2021 State of the Cloud Report, p. 41

[13] Flexera 2022 State of the Cloud Report, p. 44

[14] David Chou, "Strong User Authentication on the Web," The Architecture Journal, August 2008 as available on http://msdn.microsoft.com/en-us/library/cc838351.aspx

[15] Durbin S., "Information security without boundaries," Network Security, Feb., 2011.

[16] Bertino E., Paci F., Ferrini R., "Privacy preserving Digital Identity Management for Cloud Computing," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2009.

[17] Vishal Paranjape, Vimmi Pandey, "An Improved Authentication Technique with OTP in CloudComputing," International Journal of Scientific Research in Computer Science and Engineering, Vol-1, Issue-3, pp. 22-26, June 2013.

[18] Kim J. and Hong S., "A Consolidated Authentication Model in Cloud Computing Environments," International Journal of

Multimedia and Ubiquitous Engineering, Vol. 7, No. 3, July, 2012

[19] Moghaddam F. F., Karimi O., Hajivali M., “Applying a Single Sign-On Algorithm Based on Cloud Computing Concepts for SaaS Applications,” IEEE 11th Malaysia International Conference on Communications, November 2013, Kuala Lumpur, Malaysia

[20] Farhatullah M., “ALP: An authentication and leak prediction model for Cloud Computing privacy,” 3rd IEEE International Advance Computing Conference (IACC), 2013

[21] Chaimae E., Rahal R., Abdellatif E. A., “ECC Certificate for Authentication in Cloud-Based RFID,” 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech), May, 2016

[22] Msahli M., Hammi M. T., Serhrouchni A., Safe box cloud authentication using TLS extension, International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC), August, 2015

[23] R. Amin, T. Gaber, G. ElTaweel, and A. E. Hassanien, “Biometric and traditional mobile authentication techniques: Overviews and open issues,” in Bio-Inspiring Cyber Security and Cloud Services: Trends and Innovations. Berlin, Germany: Springer, 2014, pp. 423–446

[24] S. Furnell, N. Clarke, and S. Karatzouni, “Beyond the PIN: Enhancing user authentication for mobile devices,” Comput. Fraud Security, vol. 2008, no. 8, pp. 12–17, 2008.

[25] Y. Yang, G. D. Clark, J. Lindqvist, and A. Oulasvirta, “Free-form gesture authentication in the wild,” in Proc. CHI Conf. Human Factors Comput. Syst., 2016, pp. 3722–3735.

[26] Z. Zhao, G.-J. Ahn, and H. Hu, “Picture gesture authentication: Empirical analysis, automated attacks, and scheme evaluation,” ACM Trans. Inf. System Security, vol. 17, no. 4, p. 14, 2015.

[27] D. Nyang et al., “Two-thumbs-up: Physical protection for pin entry secure against recording attacks,” Computer Security, vol. 78, pp. 1–15, Sep. 2018

[28] Deutschmann I., Nordström P., Nilsson L., “Continuous Authentication Using Behavioral Biometrics,” IT Professional, Vol 15, Issue 4, pp. 12 - 15, 2013.

[29] P. S. Teh, A. B. J. Teoh, and S. Yue, “A survey of keystroke dynamics biometrics,” The Scientific World J., vol. 2013, Nov. 2013, Art. no. 408280.

[30] J. Wu and Z. Chen, “An implicit identity authentication system considering changes of gesture based on keystroke behaviors,” Int. Journal of Distributed Sensor Networks, vol. 11, no. 6, pp. 470274:1–470274:16, 2015.

[31] M. N. Aman, M. H. Basheer, and B. Sikdar, “Two-factor authentication for IoT with location information,” IEEE Internet Things J., vol. 6, no. 2, pp. 3335–3351, Apr. 2019.

[32] Y. Liang, S. Samtani, B. Guo, and Z. Yu, “Behavioral Biometrics for Continuous Authentication in the Internet-of-Things Era: An Artificial Intelligence Perspective,” IEEE Internet Things J., vol. 7, no. 9, pp. 9128–9143, 2020, doi: 10.1109/JIOT.2020.3004077.

[33] G. Bansal, N. Naren, V. Chamola, B. Sikdar, N. Kumar, and M. Guizani, “Lightweight Mutual Authentication Protocol for V2G Using Physical Unclonable Function,” IEEE Trans. Veh. Technol., vol. 69, no. 7, pp. 7234–7246, 2020, doi: 10.1109/TVT.2020.2976960.

[34] R. Shashidhara, S. K. Nayak, A. K. Das, and Y. Park, “On the Design of Lightweight and Secure Mutual Authentication System for Global Roaming in Resource-Limited Mobility Networks,” IEEE Access, vol. 9, pp. 12879–12895, 2021, doi: 10.1109/ACCESS.2021.3050402.

[35] R. Ferrero, F. Gandino, B. Montrucchio, M. Rebaudengo, A. Velasco, and I. Benkhelifa, “On gait recognition with smart[1]phone accelerometer,” in Proceedings of 4th Mediterranean Conf. Embedded Comput. (MECO), 2015, pp. 368–373.

[36] A. L. Fantana, S. Ramachandran, C. H. Schunck, and M. Talamo, “Movement based biometric authentication with smartphones,” in Proc. Int. Carnahan Conf. Security Technol. (ICCST), 2015, pp. 235–239

[37] Schneier B., Applied Cryptography, John Wiley & Sons (Asia) Pte Ltd, ISBN 9971-51-348-X.

[38] Insaf Adjabi, Abdeldjalil Ouahabi, Amir Benzaoui, Abdelmalik Taleb-Ahmed, Past, Present, and Future of Face Recognition: A Review, Electronics 2020, 9, 1188; doi:10.3390/electronics9081188

[39] Akshay A. Pawle, Vrushsen P. Pawar, “Face Recognition System (FRS) on Cloud Computing for User Authentication,” International Journal of Soft Computing and Engineering (IJSCE), Volume-3, Issue-4, September 2013.

[40] Hua-Hong Zhu, Qian-Hua He, Hua-Hong Zhu, Hong Tang, Wei-Hua Cao, Voiceprint-Biometric Template Design and Authentication Based on Cloud Computing Security, IEEE International Conference on Cloud and Service Computing, 2011

[41] Chen Wang, Yan Wang, Yingying Chen, Hongbo Liu, Jian Liu, User authentication on mobile devices: Approaches, threats and trends, Computer Networks, Volume 170, 2020, 107118, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2020.107118.

[42] Information Classification theory as available on http://en.wikipedia.org/wiki/Classified_information

[43] Suri P. R., Deora S. S., “A cipher based on 3D Array Block Rotation,” International Journal of Computer Science and Network Security, Vol. 10, No. 2, pp. 186-191, Feb., 2010.

[44] Suri P. R., Deora S. S., “3D Array Block Rotation cipher: an improvement using shift,” Global Journal of Computer Science and Technology, Vol. 11, Issue 19, pp. 17-23, Version 1.0, November, 2011.

[45] Mandylion Research Labs accessed from http://www.mandylionlabs.com/index15.htm] on March, 2017.