Ingénierie des Systèmes d'Information
Vol. 28, No. 6, December, 2023, pp. 1669-1678
Journal homepage: http://iieta.org/journals/isi

Advancing Secure Mobile Cloud Computing: A Chaotic Maps-Based Password Key Agreement Protocol

Syed Shakeel Hashmi[1*], Arif Mohammad Abdul[2], Arshad Ahmad Khan Mohammad[2], C. Atheeq[2], Ravi Chinapaga[3]

[1] Department of Electronics and Communication Engineering, Faculty of Science and Technology (IcfaiTech), The ICFAI Foundation for Higher Education (Deemed to be University), Hyderabad 501203, India
[2] Department of CSE, GITAM Deemed to be University, Hyderabad 502329, India
[3] Department of CSE, AVN Institute of Engineering and Technology, Hyderabad 501510, India

Corresponding Author Email: hashmi@ifheindia.org

https://doi.org/10.18280/isi.280625

Received: 11 August 2023
Revised: 24 November 2023
Accepted: 4 December 2023
Available online: 23 December 2023

Keywords:
security, authentication, key agreement, Mobile Cloud Computing (MCC), AVISPA Chaotic Maps, mutual authentication

ABSTRACT

The exponential growth in mobile technology has precipitated a substantial increase in global IP traffic, predominantly fueled by mobile devices. Mobile Cloud Computing (MCC) emerges as a viable solution to the inherent resource limitations of these devices, yet the security of data access remains a paramount concern, particularly in the context of dynamic user behavior. This paper introduces an innovative password-based authenticated key exchange protocol tailored for secure communication within MCC frameworks. Existing solutions, while addressing several challenges of MCC, fall short in adequately tackling issues related to dynamic user behavior and resource constraints. The proposed protocol is designed to address these specific deficiencies, thereby enhancing the security of data access in MCC environments. Employing Chaotic Maps for protocol resilience, symmetric encipherment for robust data protection, and one-way hash functions to bolster the security framework, this protocol is rigorously evaluated using the AVISPA tool. The results demonstrate that the protocol offers superior security and efficiency, and exhibits enhanced resilience against a spectrum of attacks compared to existing schemes. A thorough analysis is conducted to evaluate the protocol's defenses against insider threats, replay attacks, and other potential vulnerabilities, providing a comprehensive understanding of its robust security features. Conclusively, this protocol establishes a secure paradigm for key agreements in MCC, outperforming existing schemes with a significant reduction in execution time by up to 60%, marking a notable advancement in the realm of MCC security.

1. INTRODUCTION

MCC, an integration of cloud computing and mobile communication technologies [1], empowers mobile users to access computing resources from the cloud. Cloud computing offers a range of services to surmount the inherent limitations of mobile devices, such as battery energy, memory, and computing power. This integration effectively addresses resource constraints on mobile devices by harnessing the capabilities of cloud computing in conjunction with mobile communication.

A paramount challenge in MCC is securing sensitive data outsourced to the cloud, given its open platform nature and untrusted storage environment. Encrypting outsourced data is a prevalent approach to bolster security, preventing unauthorized data access [2]. Several mechanisms for data encryption prior to cloud outsourcing have been proposed in the literature [2]. In these models, clients encrypt data before outsourcing, while users retrieve and decrypt the encrypted data upon access. These schemes generally presume the users to be reputable and authorized for data retrieval. However, this assumption engenders security challenges in distributed computing environments like MCC, where users dynamically access cloud services via public networks, thus compromising authenticity and access control.

The principles of Confidentiality, Authentication, and Authorization (CAA) [3] are vital in establishing robust security measures in MCC. Authentication and access control, coupled with data secrecy, are fundamental requirements for secure operations. Confidentiality is maintained through encryption methods applied to data before cloud storage, safeguarding sensitive information from unauthorized entities. Users are authenticated through a reliable system, verifying their credentials. Authorization, on the other hand, governs user interaction with cloud services based on assigned permissions, ensuring controlled access to cloud resources.

The exigency for secure communication and information exchange is paramount in today's digital landscape. The proposed study posits that data confidentiality [4] is ensured by storing data in an encrypted format within the Cloud Computing (CC) environment, complemented by a trust-based access control mechanism for authentication [5]. However, confidentiality and authorization alone are insufficient for comprehensive security. Authentication, which verifies the identities of the communicating entities, is equally critical. Therefore, the objective is to provide robust authentication between computing entities, necessitating mechanisms that are congruent with the resource constraints of the environment.

This research introduces a password-based authenticated key exchange protocol specifically designed for cloud computing environments. Mutual authenticated key agreement stands as a fundamental, straightforward, and indispensable approach to securing any communication environment. This process involves two parties mutually authenticating and subsequently agreeing upon a secret session key, derived through a process of information negotiation. Such an approach is essential to safeguard the communication environment from unauthorized access, ensuring that the session key is securely established solely between legitimate entities.

Existing mutual authentication protocols [6-14] have been scrutinized, revealing vulnerabilities to various attacks, including man-in-the-middle, replay, and denial-of-service attacks. A notable drawback of these protocols is their considerable computational overhead, predominantly stemming from complex cryptographic operations. This results in performance limitations, especially in mobile cloud environments where processing power and network resources are inherently constrained.

The utilization of unreliable or slow networks for connecting mobile devices to the internet can exacerbate the performance challenges posed by the computational demands of authentication protocols. This issue is particularly pronounced in mobile cloud environments where cloud servers may be geographically distant from the mobile devices, leading to significant network latency. These challenges are crucial considerations in the development of an effective mutual authentication protocol for mobile cloud environments.

Jegadeesan et al. [15] have underscored critical shortcomings in existing authentication schemes within MCC, including vulnerabilities to user impersonation, mutual authentication failures, and susceptibility to Man-in-the-Middle (MITM) attacks. In response, this paper introduces an efficient password-based authenticated key exchange protocol tailored for Cloud Computing environments. The protocol employs Chaotic Maps, based on the Discrete Logarithmic and Diffie-Hellman problems, to facilitate secure session key exchange and mutual authentication in MCC. To bolster security, symmetric encipherment and one-way hash functions are integrated into the protocol, providing robust defenses against replay and MITM attacks. The method is specifically designed to manage identity authentication and securely exchange session keys between computing entities in MCC environments.

The structure of the paper is as follows: the subsequent section delineates related work in this field. Section 3 presents the proposed protocol in detail, while Section 4 is dedicated to the performance evaluation of the protocol. The paper concludes with a discussion of the results and final conclusions.

2. LITERATURE SURVEY

Cloud computing is recognized as a distributed computing environment that epitomizes a ubiquitous model, characterized by a dynamically adjustable pool of resources that are readily accessible and utilizable. Salient features of cloud computing include its high scalability, elasticity, and the provision of on-demand services at reduced costs [16]. Additionally, the cloud offers resource services to users on a pay-per-use basis, thereby minimizing management efforts and effectively addressing the limitations associated with utility computing.

MCC represents a paradigm shift in computing, whereby mobile devices leverage cloud resources. MCC synthesizes two significant technological advancements: cloud computing and mobile communication. In this framework, cloud computing offers a suite of services that alleviate the constraints of resource-limited devices, such as limited battery life, memory, and processing power. The amalgamation of cloud computing with mobile communication effectively resolves the resource constraint issues inherent in mobile devices.

MCC is employed to deliver hosted resources over public networks. A myriad of services, including Platform as a Service, Software as a Service, and Infrastructure as a Service, are now accessible via the cloud. This on-demand provision of resources, coupled with cost-effectiveness, has led to a shift in organizational security perimeters, consequently elevating the risk of security breaches. Ensuring data security in the cloud, particularly in terms of confidentiality and authentication, becomes paramount. Encrypting data stored in the cloud facilitates confidentiality. However, in an era of escalating information thefts, cloud service providers must exercise heightened vigilance when granting access to cloud services. Mutual authentication between users and cloud services is imperative to mitigate unlawful activities in MCC. In implementing authentication, it is crucial that the user's identity is protected from cloud service providers. Moreover, any proposed solution must be tailored to accommodate low-end devices and be computationally and storage-efficient, as mobile devices with limited resources cannot sustain a heavy computational load.

In the realm of authentication protocols, numerous RSA (Rivest-Shamir-Adleman)-based methods have been explored, but their computational intensiveness renders them inefficient for MCC applications [17]. As an alternative, various protocols leveraging Elliptic Curve Cryptography (ECC) [18-21] have been developed, noted for their lower computational overhead compared to RSA [22]. Nonetheless, these ECC-based protocols often involve third parties to facilitate authentication between mobile users and cloud servers, leading to additional computational burden [23].

An analysis of these ECC-based authentication schemes concluded that their high computational overhead detracts from their practicality and fails to adequately protect the user's identity from cloud service providers. Therefore, there is a need for an authenticated key agreement protocol that is not only computationally efficient but also ensures the protection of the user's identity from cloud service providers. Such a protocol should allow the secret key to be agreed upon exclusively between the intended parties.

Existing mutual authentication protocols have been scrutinized and found to be vulnerable to various attacks, including man-in-the-middle, replay, and denial-of-service attacks, as delineated in Table 1. These vulnerabilities compromise the protocols' ability to provide robust mutual authentication, leaving data at risk of unauthorized access. A notable issue with these protocols is their considerable computational overhead, primarily stemming from complex cryptographic operations, which can result in significant delays and decreased system performance. This challenge is exacerbated in mobile cloud environments, where mobile devices are often limited in processing power, memory, and battery life.

Additionally, the use of unreliable or slow networks for connecting mobile devices to the internet can further compound the performance issues associated with the computational demands of authentication protocols. This becomes a significant concern when considering the geographical distance between mobile devices and cloud servers, which can lead to high network latency. Collectively, these factors underscore the substantial challenges faced in developing an efficient mutual authentication protocol tailored for mobile cloud environments.

Table 1. Comparison of existing password-based mutual authentication protocols

Protocol	Challenges Addressed	Drawbacks/Issues
Goh [16]	The nonce mechanism used to prevent replay attacks may be vulnerable to MITM attacks [15]	Vulnerable to PIA (Password Impersonation Attack), UA (Unauthorized Access), KPTA (Key Pair Theft Attack), cannot provide perfect FS
Tan et al. [17]	Goh [16]	Spoofing attack, no single registration, unauthorized login, no password change facility
Slocum [18]	Tan et al. [17]	Vulnerable to UIA, session-specific temporary info attack, SSCA (Session-Specific Temporary Info Attack), password and IGA
Song et al. [19]	Slocum [18]	Improved chaotic map based MSA, Vulnerable to UIA, RA, does not have user revocation, no feasibility of updating user info
Curtmola et al. [21]	Vulnerable to Server-Impersonation Attack (SIA), UA property was not provided, ephemeral secrets leakage attack and MA cannot be achieved [20].	Vulnerable to UIA (User Impersonation Attack), DOSA (Denial-of-Service Attack), RA (Replay Attack), no feasibility of updating user info, does not address UR
Abdul et al. [23]	PIA, UIA, SSCA and server cannot differentiate between the two logins [22]	Improved MSA protocol, Vulnerable to PIA, UIA, SSCA, server cannot differentiate between the two logins

Recent analysis [24] of a MCC authentication scheme identified critical deficiencies. Firstly, the scheme lacks protection against user impersonation attacks, allowing attackers to easily impersonate legitimate users and gain unauthorized system access. Secondly, it fails to provide mutual authentication, a vital security feature, leaving both users and service providers unable to verify each other's identities. Finally, the scheme is vulnerable to Man-in-the-Middle (MITM) attacks, whereby an attacker can intercept and alter communications between the user and the service provider without detection. These shortcomings highlight the need for a more robust and secure authentication framework in MCC.

The impetus for this research is rooted in the vulnerabilities and performance challenges identified in existing mutual authentication protocols within mobile cloud environments. Our comprehensive analysis has uncovered several critical vulnerabilities, such as susceptibility to MITM, replay, and denial-of-service attacks. These weaknesses render the protocols inadequate for ensuring robust mutual authentication, leaving data vulnerable to unauthorized access. Additionally, these protocols are plagued by high computational overhead, leading to significant delays and diminished system performance. This issue is particularly acute in mobile devices, constrained by limited processing power, memory, and battery life. The situation is further exacerbated when unreliable or slow networks are used in mobile cloud environments, presenting substantial challenges in developing an effective mutual authentication protocol. Recent research has also underscored existing schemes' limitations, including their inability to prevent user impersonation attacks, failure to provide mutual authentication, and vulnerability to MITM attacks. Consequently, our research aims to overcome these limitations by proposing a new mutual authentication protocol tailored for mobile cloud environments.

In response to these challenges and gaps, this work introduces a novel password-based authenticated key exchange protocol, specifically designed for Cloud Computing environments. Utilizing Chaotic Maps, this protocol addresses the Discrete Logarithmic and Diffie-Hellman problems. Implemented within the MCC framework, it not only enables secure session key exchange but also assures mutual authentication. The protocol's integrity is further enhanced by incorporating symmetric encipherment and a one-way hash function, which effectively mitigate replay and MITM attacks. This authentication mechanism's strength is intrinsically linked to its key management [25], ensuring the secure transfer of the session key between entities through mutual authentication, thereby safeguarding the computing environment.

3. PROPOSED MODEL

The Cloud computing environment is composed of a cloud service provider (CSP), client, and users. Here, the client outsources the resources to the cloud. CSP offers a data storage service and dynamically allocates computational resources to the users and client. So, the client stores its data in the cloud, and users retrieve the stored data from the cloud.

Consider the cloud computing environment with CSP, client, and users, where the client stored the collection of n data files, say, $[F_1, F_2, ..., F_n]$ to cloud in an encrypted form to enable confidentiality of the data. The outsourced files must be stored and controlled so that authenticated users only access them. Thus, work designed the password-based authentication key agreement-based protocol with the help of Chaotic Maps-based Discrete logarithmic problem and Diffie-Hellman problem to accomplish the mutual authentication between CSP and users. The explanation of the proposed protocol is as follows:

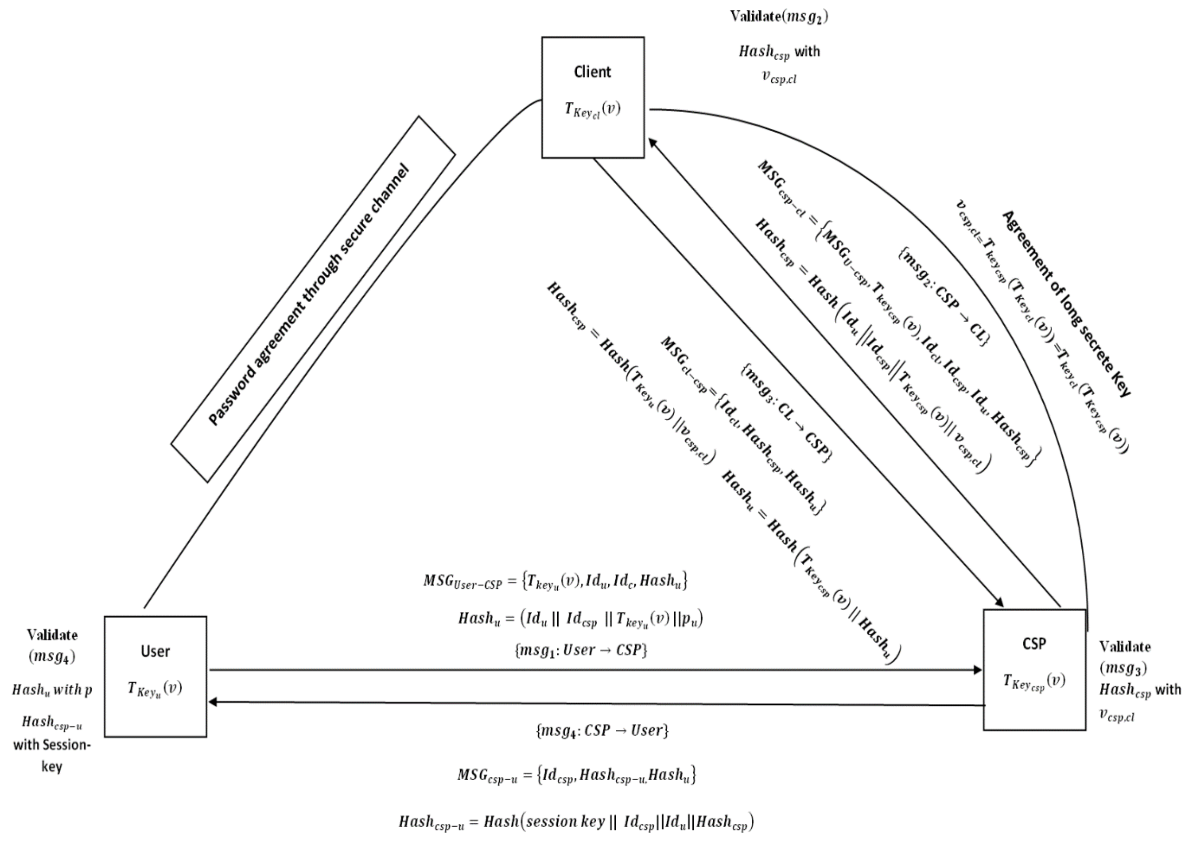

Validate(msg_2)
$Hash_{csp}$ with
$v_{csp.cl}$
Client
$T_{Key_{cl}}(v)$
Password agreement through secure channel
$MSG_{csp-cl} = \{MSG_{U-csp}, T_{key_{csp}}(v), Id_{cl}, Id_{csp}, Id_u, Hash_{csp}\}$
$Hash_{csp} = Hash(Id_u||Id_{csp}||T_{key_{csp}}(v)||v_{csp.cl})$
$\{msg_2: CSP \to CL\}$
Agreement of long secrete Key
$v_{csp.cl} = T_{key_{csp}}(T_{Key_{cl}}(v)) = T_{key_{cl}}(T_{Key_{csp}}(v))$
$Hash_{csp} = Hash(T_{Key_u}(v)||v_{csp.cl})$
$MSG_{cl-csp} = \{Id_{cl}, Hash_{csp}, Hash_u\}$
$\{msg_3: CL \to CSP\}$
$Hash_u = Hash(T_{Key_{csp}}(v)||Hash_u)$
$MSG_{User-CSP} = \{T_{key_u}(v), Id_u, Id_c, Hash_u\}$
$Hash_u = (Id_u || Id_{csp} || T_{key_u}(v)||p_u)$
$\{msg_1: User \to CSP\}$
Validate (msg_4)
$Hash_u$ with p
$Hash_{csp-u}$ with Session-key
User
$T_{Key_u}(v)$
CSP
$T_{Key_{csp}}(v)$
Validate (msg_3)
$Hash_{csp}$ with
$v_{csp.cl}$
$\{msg_4: CSP \to User\}$
$MSG_{csp-u} = \{Id_{csp}, Hash_{csp-u}, Hash_u\}$
$Hash_{csp-u} = Hash(session\ key || Id_{csp}||Id_u||Hash_{csp})$

Figure 1. PBMAK architecture

User, client, and cloud service provider (CSP) are entities involved in the cloud computing environment. Whenever a user wants to access the data from the cloud, the User must authenticate to access the system. The proposed protocol considers that before establishing communication, the User and client agree on a password over an insecure channel. In contrast, the CSP and client agree on a long session key with the help of the Chaotic Maps-based Diffie-Hellman problem. Here, the client acts as a middleman and authenticates the User and CSP by verifying their identities. The user and CSP authenticate the client by verifying the client's identity. After that, the User and CSP verify each other and continue to authenticate to establish a secret key.

3.1 Mutual authenticated key agreement algorithm

The proposed algorithm Password-Based Mutual Authenticated Key Agreement for Cloud computing (PBMAK) is designed by considering the following considerations:

• A unique identifier is assigned to all the participating entities in the authentication process, say (Id_i).

• The participating entities agree to process the authentication using the Chebyshev polynomial Chaotic Maps approach and a one-way hash function denoted as $H(\cdot)$, a pair of symmetric-key encryption/decryption functions $E_k(\ )/D_k(\ )$ with key k, and random number 'v' to compute the Chebyshev polynomial Chaotic Maps operation.

Figure 1 shows the architecture of PBMAK, consisting of cloud users, cloud service providers (CSP), and Cloud clients. The client establishes communication between cloud users and cloud service providers. When users approach the Cloud via Client and CSP, the client maintains all users' credentials as the hash value to avoid insider attacks. Whenever a user wants to use the cloud services, the User provides their credentials to CSP; CSP cannot verify the User, so forwards to the client for user verification. After the client validates the User and sends it back to CSP, CSP generates a session key to establish communication with the cloud user. Once the connection is established, data exchange will take place.

3.2 Authentication process to get the cloud services

To access cloud services, the User must provide credentials against the CSP. After verification of the User, the User becomes an authenticated cloud user. The system has gone through the process of authenticating the User. In this process, the cloud user, CSP, and client agree on some value, 'v'. (random number) Initially, the User registers with the client to access the cloud services. In the registration process, the User selects '*id*', password '*p*' and secret key '*key*'. Compute '*public key*' with Chebyshev's polynomial Chaotic Maps approach along with agreed value 'v', and named as $\boldsymbol{T_{Key_u}(v)}$, using equation (2). Also, the User generates the hash value of the chosen password ***Hash***(*p*), i.e., $\boldsymbol{Hash_p}$ assumes that this password **[**$\boldsymbol{Hash_p}$**]** communicates with the client through a secure channel with the corresponding chosen '*id*'. Establishing a secure channel for agreeing on a password is either a physical meeting or encrypted communications. The client stores the user password [$\boldsymbol{Hash_p}$] associated with '*id*' for future verification. Even the client is also unaware of the User's correct password. To communicate with CSP, the client chooses a secret key '*key*' and computes the '*public key*' with Chebyshev's polynomial Chaotic Maps approach along with agreed value $'v'$, and named as $\boldsymbol{T_{Key_{cl}}(v)}$. Similarly, CSP also chooses a secret key '*key*' and computes this key '*public key*' with Chebyshev's polynomial Chaotic Maps approach along with agreed value 'v', and named as $\boldsymbol{T_{Key_{csp}}(v)}$.

Table 2. Notation and description

S.NO	Notation	Abbreviations	Description
1	User	u	Entity representing the User or the authenticating entity.
2	Client	cl	Entity representing the client or the service provider
3	Cloud service provider	csp	Entity representing the cloud service provider
4	Password	p	During registration all the users agree on the password
5	Secret Key	key' $T_{key_u} = User\ Secret\ key$ $T_{key_{csp}} = CSP\ Secret\ key$	Private key known only to the respective entity
6	Agreed Value	V	Common value agreed upon by the entities for computation
7	Variable	X	Variable used in the polynomial function for chaos mapping
8	Chebyshev's polynomial Chaotic Maps approach	$T_x(v) = 2v * T_{x-1}(v) - T_{x-2}(v) * (mod\ X)$	Authentication approach utilizing Chebyshev's polynomial Chaotic Maps
9	Public Key	$T_{Key_i}(v)$ i=u, cl, csp	Computed public key using the secret key and agreed value
10	Semigroup property	$T_a(T_b(v)) = T_b(T_a(v))$	Property of an algebraic structure where the operation is associative and closed

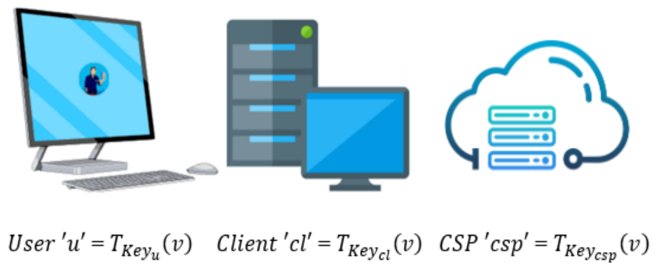

$User\ 'u' = T_{Key_u}(v)$
$Client\ 'cl' = T_{Key_{cl}}(v)$
$CSP\ 'csp' = T_{Key_{csp}}(v)$

Figure 2. Authentication parameters in Chebyshev Polynomial form

Table 2 provides abbreviations, notations, and descriptions used in the authentication algorithm. Figure 2 discusses the users, clients, and cloud service provider computed public keys, i.e., as $T_{Key_u}(v)$, $T_{Key_{cl}}(v), T_{Key_{csp}}(v)$ with their corresponding secret key '*key*'. These public keys are computed with a commonly agreed value '*v*' with Chebyshev's polynomial Chaotic Maps approach. If the adversary tries to understand these values but is difficult to decode because ['*key*'] is unknown. Here, this proposed approach restricts the man-in-the-middle attack.

Our proposed key agreement method leverages Chaotic Maps, utilizing Chebyshev polynomials from chaos theory—a mathematical field studying highly sensitive dynamic systems. Chebyshev polynomials are defined as follows:

$$\begin{aligned}
&\cos(n\theta)\text{ could be written in the polynomial of }\cos(\theta)\\
&\cos(n\theta) = T_n * \cos(\theta)\\
&\cos((n+1)*\theta) = 2*\cos(n\theta)*\cos(\theta) - \cos((n-1)*\theta)\\
&T_{n+1}\cos(\theta) = 2*T_n\cos(\theta)*\cos(\theta) - T_{n-1}\\
&\cos(\theta)T_{n+1}(x) = 2*x*T_n(x) - T_{n-1}(x)
\end{aligned}$$

Let,

$$T_x(v) = \cos(x\arccos(v)) \tag{1}$$

where, *x=integer*, *v=variable*.

Chebyshev's polynomial Chaotic Maps approach T_x: *x* in Chebyshev's polynomial is defined as in recurrence relation is:

$$T_x(v) = 2v*T_{x-1}(v) - T_{x-2}(v)*(\text{mod } X)\ x \geq 1 \tag{2}$$

where, *x*=0, 1, 2, 3, ……… i.e., $x \in (-\infty, \infty)$, $T_0(v)$=1, $T_1(v)$=v, ….

Chebyshev's polynomial Chaotic Maps approach one of the most important properties is semigroup property, i.e.:

$$T_a(T_b(v)) = T_b(T_a(v)) \tag{3}$$

Based on equation (3), equation (4) is constructed for the mutual authentication between User and CSP as:

$$v_{u,csp} = T_{key_u}\left(T_{\text{Keycsp}}(v)\right) = T_{\text{keycsp}}\left(T_{\text{Keyu}}(v)\right) \tag{4}$$

where, *v*=Agreed value, T_{key_u}=User Secret key, $T_{key_{csp}}$=CSP Secret key, $v_{u,csp}$=User and CSP computed value.

After the registration process between User and client, the User can now establish a connection between the User and CSP. After generating a session-key user can send the message-1 to CSP {*msg*$_1$: *User*→*CSP*}.

$$\begin{aligned}
&MSG_{User-CSP} = \left\{T_{\text{key}_u}(v), Id_u, I_{csp}, \text{ Hash}_u\right\}\\
&\text{Hash}_u = \left(\text{Id}_u \| Id_{csp} \| T_{\text{key}_u}(v) \| \text{Hash}_p\right)
\end{aligned} \tag{5}$$

where, $T_{key_u}(v)$=Computed public value of user, *Id*$_u$=Identity of a user, *Id*$_{csp}$=Identity of cloud service provider, *Hash*$_u$=Digested information of a user including hash password.

When a user needs the Cloud services, send a message to CSP. The message contains the User's computed public key [$T_{key_u}(v)$] using his private key '*key*', identity [*Id*$_u$], particular identity [*Id*$_{csp}$] of CSP to which the User wants to communicate and digest message [*Hash*$_u$].

To provide secrecy from adversaries and CSP, users hash the values like [*Id*$_u$, *Id*$_{csp}$, $T_{key_u}(v)$ and *Hash*$_p$ into *Hash*$_u$] and send to CSP. After receiving *msg*$_1$ CSP redirects to the client for validation because CSP does not know the User's password for cross verification.

To establish mutual authentication between CSP and the client based on the semigroup property of Chebyshev's polynomial Chaotic Maps approach with equation (3), they form equation (6):

$$v_{csp,cl=}T_{key_{csp}}\left(T_{Key_{cl}}(v)\right) = T_{key_{cl}}\left(T_{Key_{csp}}(v)\right) \tag{6}$$

where, *v*=Agreed value, $T_{key_{cl}}$ =Client Secret key,

$T_{key_{csp}}$ =CSP Secret key, $v_{csp,cl}$=CSP and Client computed value.

After computing the session key between CSP and client, CSP can now establish a connection between CSP and client. Message-2 sent by CSP to client {*msg*₂: *CSP→CL*}.

$$\begin{aligned} &MSG_{csp-cl}=\left\{MSG_{User-csp},T_{key_{csp}}(v),Id_{csp},Id_u,Id_{cl},Hash_{csp}\right\}\\ &\text{Hash}_{csp}=\text{Hash}\left(Id_u \parallel Id_{csp}\left\|T_{Key,ycs}(v)\right\|v_{csp,cl}\right)\end{aligned} \tag{7}$$

where, $T_{key_{csp}}(v)$=Computed public key of CSP, Id_u=Identity of a user, Id_{csp}=Identity of cloud service provider, Id_{cl}=Identity of a client, $Hash_{csp}$=Digested information from CSP.

CSP redirects the received msg_1 from a user and generate the msg_2, which contains the user's computed public key [$T_{key_u}(v)$] using his private key '*key*', *CSP*'s computed public key [$T_{key_{csp}}(v)$] using his private key '*key*', identity [Id_u], particular identity [Id_{csp}] of CSP to which the User wants to communicate, particular identity [Id_{cl}] of the client to which the CSP wants to communicate and digest message [$Hash_u$] from a user, digest message [$Hash_{csp}$] from a CSP. For message integrity, CSP hash the values like [Id_u, Id_{csp}, $T_{key_{csp}}(v)$ and $v_{csp,cl}$ into $Hash_{csp}$] and send it to the client.

Client computes the $Hash_{cl}$ based on the information of msg_2 and compares the hash values: $Hash_{cl}$== $Hash_{csp}$.

If both are equal, then it validates msg_1 by computing $Hash_u$ using stored digested password $Hash_p$ which is associated with Id_u, then validate by comparing computed $Hash_u$ with received [$Hash_u$] from a CSP.

$$\text{Hash}_u(CSP)==\text{Hash}_u(\textit{from Client})$$

If validation is successful, then the client sends a message-3 to CSP {*Msg*₃: *CL→CSP*}.

$$\begin{aligned} &MSG_{cl-csp}=\left\{Id_{cl},\text{Hash}_{csp},\text{Hash}_u\right\}\\ &\text{Hash}_{csp}=\text{Hash}\left(Id_u \parallel Id_{csp}\left\|T_{\text{Key}_{csp}}(v)\right\|v_{csp,cl}\right)\\ &\text{Hash}_u=\left(Id_u\left\|Id_{csp}\right\|T_{\text{key}_u}(v)\|\text{Hash}_p\right)\end{aligned} \tag{8}$$

where, Id_{cl}=Identity of a client, $Hash_{csp}$=Digested information from Client, $Hash_u$=Digested information of a user including hash password.

If validation is unsuccessful, then the client sends a message-3 to CSP {*msg*₃: CL→CSP}.

$$MSG_{cl-csp}=\left\{Id_{cl},Hash_{csp}\right\} \tag{9}$$

where, Id_{cl}=Identity of a client, $Hash_{csp}$=Digested information from Client.

If validation is successful from the client, then the CSP validates msg_3 by comparing received and computed hash values.

$$\text{Hash}_{csp}==\text{Hash}_{cl}$$

If both are equal, then it validates msg_1 by comparing computed $Hash_u$ from the client with received [$Hash_u$] from a user.

$$\text{Hash}_u(User)==\text{Hash}_u(\textit{fromClient})$$

If validation is successful, then the CSP sends a message-4 to the User {*msg*₄: *CSP→User*}.

$$\begin{aligned} &MSG_{csp-u}=\left\{Id_{csp},\text{Hash}_{csp-u}\ \text{Hash}_u\right\}\\ &\text{Hash}_{csp-u}=\text{Hash}\left(\text{session key}\left[v_{csp,u}\right]\parallel Id_{csp}\left\|Id_u\right\|\text{Hash}_{csp}\right)\end{aligned} \tag{10}$$

After successful validation of msg_4 received from CSP by the User and access to the cloud services.

$$v_{u,csp}=T_{key_{csp}}\left(T_{Key_u}(v)\right)=T_{key_u}\left(T_{\text{Key}_{csp}}(v)\right) \tag{11}$$

A case study shows the construction session-key between User '*u*' and Cloud Service Provider '*csp*' agrees on '*v*'.

Assume *v*=3.

The User selects the private key as a large prime number as 'T_{Key_u}'=5 and calculates the public key with the agreed value ['*v*'] using Chebyshev's polynomial Chaotic Maps approach based on Diffie Hellman key exchange.

According to Eq. (4):

T_{Key_u}=5 is a selected secret key of a user.

$T_{Key_u}(v)$=5*3=15 computed public key by a user.

Similarly, CSP selects the private key as a large prime number as '$T_{Key_{csp}}$'=7 and calculates the public key with agreed value '*v*' using Chebyshev's polynomial Chaotic Maps approach based on Diffie Hellman key exchange.

$T_{Key_{csp}}$=7 is a selected secret key of a CSP.

$T_{Key_{csp}}(v)$= 7*3=21 computed public key by CSP.

Client computes session key using $T_{key_{csp}}(T_{Key_u}(v))$ and sends $T_{key_{csp}}(v)$ to User along msg_4. After receiving msg_4, the User computes the session key by using $T_{Key_{csp}}(T_{Key_u}(v))$. Both will substitute the values in equation (4):

$v_{u,csp} = T_{key_u}(T_{Key_{csp}}(v))$=$T_{key_{csp}}(T_{Key_u}(v))$

session key=5(7*3)=7(5*3)

session key=5(21)=7(15)

session key=105=105

Resultant values are equal, and then communication is established between the User and CSP using an agreed session key.

3.3 Mutual authentication and key agreement in User–Client–CSP

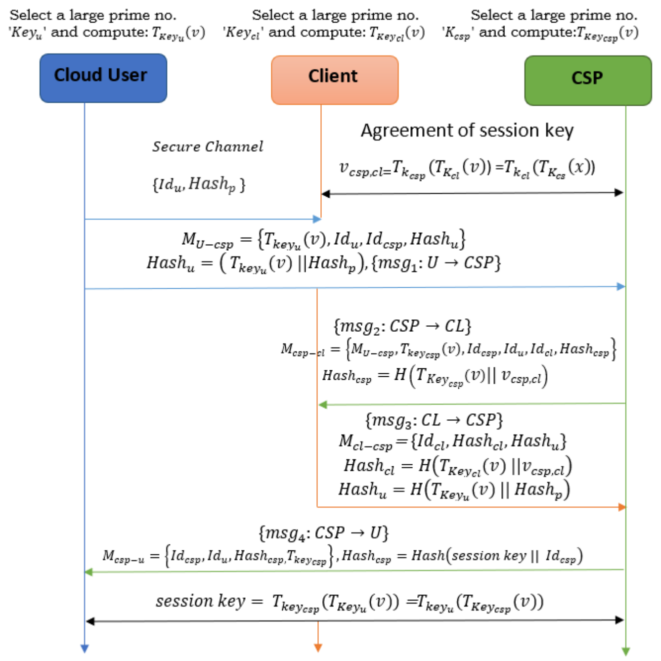
Figure 3 describes the handshake mutual authentication process in a step-by-step fashion between the User, Cloud Service Provider, and the client of the proposed scheme. The mutual authentication mechanism works on Chebyshev's polynomial Chaotic Maps approach based on the Diffie Hellman key exchange. In this approach, the registration process takes place between cloud users (*u*) and clients(*cl*). The User selects two keys: the private key and the public key.

The User selects the private key from a large prime number, i.e., 'Key_u' and computes the polynomial value by applying the recurrence relation based on Chebyshev's polynomial Chaotic Maps approach, i.e., $T_{Key_u}(v)$. '*v*' is a mutual value between cloud users(*u*), clients(*cl*) and cloud service provider (*csp*). Before starting the communication, (i) cloud users(*u*), (ii) clients(*cl*) and (iii) cloud service provider [(*csp*)] agreed on the value '*v*'.

$v =$ agreed key between user and client

$T_{\text{Key}_u}(v) =$ Public key

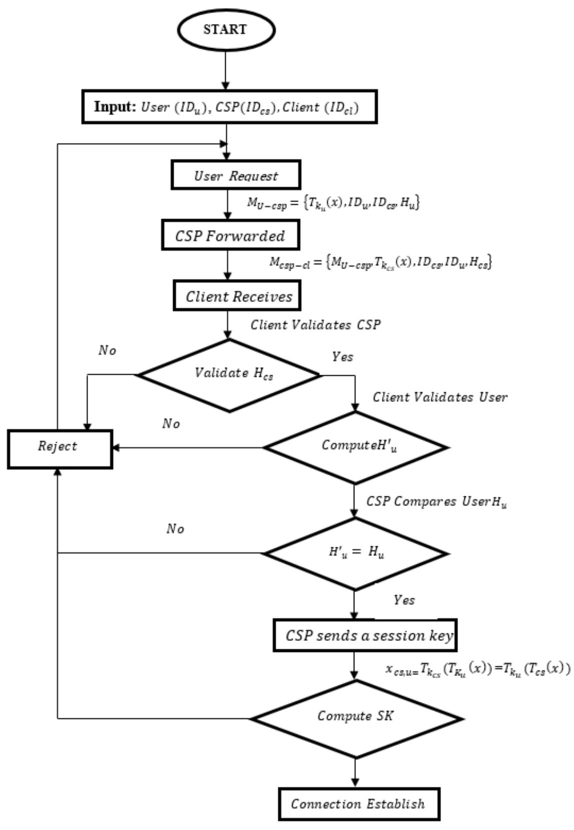
The User [(u)] selects two parameters to get into the cloud to access the services. The parameters are identity [Id_u], and password ['p']. To avoid insider attacks, users digest the password as $Hash_p$. The User {Id_u, $Hash_p$} shared through a secure channel to the client. The client stores users securely in the server database for future validation.



Select a large prime no. 'Key_u' and compute: $T_{Key_u}(v)$
Select a large prime no. 'Key_{cl}' and compute: $T_{Key_{cl}}(v)$
Select a large prime no. 'K_{csp}' and compute: $T_{Key_{csp}}(v)$
Cloud User
Client
CSP
Secure Channel
$\{Id_u, Hash_p\}$
Agreement of session key
$v_{csp,cl=}T_{k_{csp}}(T_{K_{cl}}(v)) = T_{k_{cl}}(T_{K_{cs}}(x))$
$M_{U-csp} = \{T_{key_u}(v), Id_u, Id_{csp}, Hash_u\}$
$Hash_u = (T_{key_u}(v) \| Hash_p), \{msg_1: U \rightarrow CSP\}$
$\{msg_2: CSP \rightarrow CL\}$
$M_{csp-cl} = \{M_{U-csp}, T_{key_{csp}}(v), Id_{csp}, Id_u, Id_{cl}, Hash_{csp}\}$
$Hash_{csp} = H(T_{Key_{csp}}(v) \| v_{csp,cl})$
$\{msg_3: CL \rightarrow CSP\}$
$M_{cl-csp} = \{Id_{cl}, Hash_{cl}, Hash_u\}$
$Hash_{cl} = H(T_{Key_{cl}}(v) \| v_{csp,cl})$
$Hash_u = H(T_{Key_u}(v) \| Hash_p)$
$\{msg_4: CSP \rightarrow U\}$
$M_{csp-u} = \{Id_{csp}, Id_u, Hash_{csp}, T_{key_{csp}}\}, Hash_{csp} = Hash(session\ key \| Id_{csp})$
$session\ key = T_{key_{csp}}(T_{Key_u}(v)) = T_{key_u}(T_{Key_{csp}}(v))$

Figure 3. Handshake between User, Cloud Service Provider, and Client

3.4 Mutual authentication flowchart



START
Input: $User\ (ID_u), CSP(ID_{cs}), Client\ (ID_{cl})$
User Request
$M_{U-csp} = \{T_{k_u}(x), ID_u, ID_{cs}, H_u\}$
CSP Forwarded
$M_{csp-cl} = \{M_{U-csp}, T_{k_{cs}}(x), ID_{cs}, ID_u, H_{cs}\}$
Client Receives
Client Validates CSP
No
Validate H_{cs}
Yes
Client Validates User
No
Reject
ComputeH'_u
CSP Compares UserH_u
No
$H'_u = H_u$
Yes
CSP sends a session key
$x_{cs,u=}T_{k_{cs}}(T_{K_u}(x)) = T_{k_u}(T_{cs}(x))$
Compute SK
Connection Establish

Figure 4. The flow of Mutual authentication agreements among Users, Cloud Service providers, and Client

The previous section's explanation of mutual-authenticated key agreement is represented in the flowchart, i.e., Figure 4. It shows the flow of the PBMAK algorithm; initially, the algorithm takes input as *User*(Id_u), *CSP*(Id_{csp}), *Client*(Id_{cl}), whereas *Client* is responsible for authenticating *Users* as well as *CSP*.

Client, *Users*, and *CSP* agree on one common value to generate a session key for session creation to exchange the information. In this process, users request cloud services from a cloud service provider. Before establishing a session key, they validate each other for mutual authentication; after verification, *User* and *CSP* establish a session key, and the User can access the authorized services from the cloud.

While validation, if [H'_u!=H_u] or [H_u!=H'_u], then the connection is rejected.

3.5 Mutual authentication algorithm

Algorithm.1 shows the Algorithm of the Mutual Authentication Agreement. This algorithm requires three parameters as *User*(Id_u), *CSP*(Id_{csp}), *Client*(Id_{cl}), each agrees on a value 'v' before establishing a communication path.

Algorithm 1. Mutual Authentication Agreement

Input: *User*(Id_u), *CSP*(Id_{csp}), *Hash*$_u$, *Client*(Id_{cl})
Output: Mutual Authentication
User send $M_{U-csp} = \{T_{key_u}(v), Id_u, Id_{csp}, Hash_u\}$ to CSP
CSP send $M_{csp-cl} = \{M_{U-csp}, T_{key_{csp}}(v), Id_{cs}, Hash_{csp}\}$ to CL
CL compute H'_u with user stored digested password $Hash_p$
CL send $M_{cl\text{-}csp}$={Id_{cl}, $Hash_{cl}$ = [H'_u], $Hash_u$ = [H_u]}
CSP compares H'_u=H_u
CSP Send $M_{csp\text{-}U}$= $\{Id_{csp}, Id_u, Hash_{csp}, T_{key_{cs}}\}$
CSP and User generate a session key
$v_{csp,u=}T_{key_{csp}}(T_{Key_u}(v)) = T_{key_u}(T_{key_{csp}}(v))$
If resultant values are equal
$T_{key_{csp}}(T_{Key_u}(v)) = T_{key_u}(T_{key_{csp}}(v))$
Connection establishes.
$T_{key_{csp}}(T_{Key_u}(x)) \neq T_{key_u}(T_{key_{csp}}(v))$
Reject

3.6 Result analysis

The proposed mutual authentication mechanism restricts unauthorized users and prevents insider, reply, and man-in-the-middle attacks.

3.7 Security analysis

By assuming adversaries have complete control over the transmission path. The following are the mathematical security analysis proofs of the proposed mutual authentication mechanism.

(i) Mutual Authentication over PBMAK

The PBMAK algorithm outlines a trust flow involving the User, Client, and CSP in mutual authentication. The algorithm enables users to verify the accessibility of the session key created by the client, allowing sharing without divulging a secret key.

$$v_{u,cl} = T_{key_u}\left(T_{Key_{cl}}(v)\right) = T_{key_{cl}}\left(T_u(v)\right) \quad (12)$$

The client must authenticate the User and CSP simultaneously because the client has a pivotal position in the authenticated key exchange phase. The client authenticates CSP and User by validating *Hash*$_{csp}$ with $v_{csp,cl}$ and $v_{u,cl}$.

CSP first authenticated the client by validating the *Hash*$_{csp}$ with $v_{csp,cl}$. After that, CSP will trust the User because *Hash*$_{csp}$ contains $T_{K_{csp-u}}(v)$. As for the key agreement, after authenticating each other, the User and CSP can make the key agreement.

(ii) Insider attack over PBMAK “Insider Attack attempting to gain unauthorized access”

At the time of the registration, the User selects '*id*', the password '*p*' and the secret key '*key*'. Compute '*key*' with Chebyshev's polynomial Chaotic Maps approach along with agreed value '*v*', and named as $T_{Key_u}(v)$. Also, the User generates the hash value of the chosen password *Hash*(*p*), i.e., *Hash*$_p$ assumes that this password [*Hash*$_p$] communicates with cloud client through a secure channel with corresponding chosen '*id*'. The client stores the user password [*Hash*$_p$] associated with '*id*' for future verification. Even the client is unaware of the User's correct password.

(iii) Reply attack over PBMAK “unauthorized attempts to replay previously exchanged communication between User, client, and CSP to gain illegitimate access”

Communication exchange through the secure path using User, client, and CSP created session key. Here, we assure that the sender and receiver are either User and client or User and CSP or client and CSP because CSP and User validate with $v_{csp,u}$ and $v_{u,csp}$. The client validates both by using $v_{cl,u}$ and $v_{cl,csp}$. These session keys can be created only with their agreements.

(iv) Man-in-the-middle attack over PBMAK” attacker trying to impersonate a client, CSP, or User by intercepting and manipulating the key exchange process”
An adversary tries to impersonate a client, CSP or User but not because pretending as one of them requires a secret key of the User as 'K_u', the client as 'K_{cl}', and CSP as 'K_{csp}'. These private key values are chosen from large prime values. The private key was never shared with any of them and nowhere stored; the client stores the User's password as the hash value.

4. PERFORMANCE RESULTS

The proposed work takes place in the Windows 11 operating system, Intel(R) Core (TM) i7-10610U CPU @ 1.80GHz 2.30 GHz. The operations considered for mutual authentication are ECC Scalar multiplication, point addition, one-way hash function, bilinear pairing operation, and Chebyshev's polynomial. The time for each process is calculated and presented in the Table 2 [24].

Based on the Table 2, execution time compares the efficiency of PBMAK with the existing mutual authentication schemes.

Table 2. List of operations with execution time in milliseconds

Notation	Name of Operation	Time of Operation (ms)
$T_{ECC\text{-}sm}$	ECC Scalar multiplication	42
T_{PA}	point addition	0.1
T_h	one-way hash function	0.5
T_{BP}	bilinear pairing	262
T_{CP}	Chebyshev's polynomial	21.02

Table 3. Comparison of execution time in milliseconds with various schemes

Schemes	Operations	Execution Time (ms)
Sun et al. [26]	$7T_{ECC\text{-}sm}+2T_{BP}$	818.12
Tsai and Lo [27]	$10T_{ECC\text{-}sm}+2T_{PA}$	419.863
Jegadeesan et al. [28]	$8T_{ECC\text{-}sm}+4T_h+4T_{PA}$	336.143
PBMAK	$4T_{CP}+5T_h$	84.08

Table 3 list the operations performed for mutual authentication between User and client or user and CSP or CSP and Client. While comparing the execution time of the proposed algorithm [PBMAK] with various schemes, it is identified that the PBMAK algorithm computes in less time.

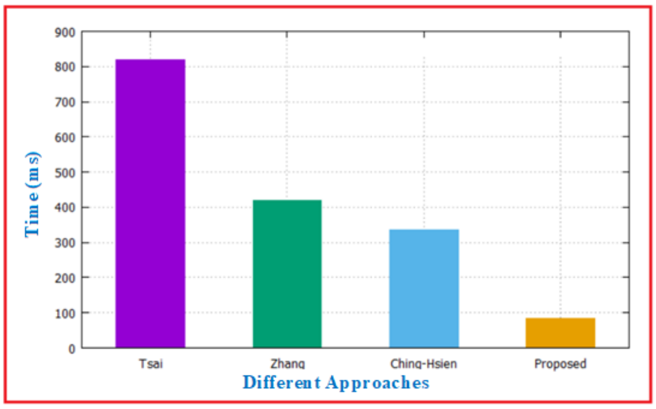


Figure 5. Execution time (ms) between different approaches

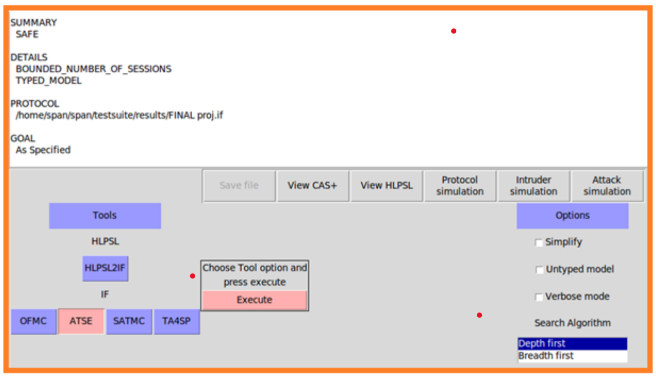
Figure 5 clearly shows PBMAK algorithm produces better execution time compared to existing algorithms. The proposed algorithm's execution time is 84.08 ms, whereas Yin et al. [25] algorithm executes in 818.12 ms, Sun et al. [26] algorithm run in 419.863 ms, and Tsai and Lo [27] take execution time in 336.143 ms.

4.1 Strength of PBMAK

AVISPA tool is used to evaluate the security analysis of the protocol. The AVISPA tool is a collection of applications for creating and analyzing formal security protocol models. It uses HLPSL (High-Level Protocol Specification Language) to generate protocol models [25]. The HLPSL is a language used by AVISPA to illustrate security protocols and identify their intended security attributes, as well as a collection of tools to test them formally. HLPSL is a Role-Based Language; the SPAN tool executes the algorithm. The SPAN tool, integral to HLPSL, executes the algorithm. In this context, the SPAN tool plays a crucial role in the formal testing process.

The proposed protocol is implemented in the AVISPA tool to verify its security strength; the results are shown in Figure 6. The choice of the AVISPA tool is justified by its comprehensive capabilities in creating and analyzing formal security protocol models. Its use of HLPSL aligns with the need for a high-level language for protocol specification, and the inclusion of the SPAN tool enhances the formal testing process.

To implement the proposed protocol in the AVISPA tool, security strength parameters were defined through a meticulous process, considering key aspects of the PBMAK protocol. Vulnerability tests were conducted, evaluating the protocol's resilience against various security threats and verifying its effectiveness in achieving the intended security attributes.



```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/FINAL proj.if

GOAL
  As Specified
```

Figure 6. Simulation result of mutual authentication algorithm

5. CONCLUSIONS

Paper presents a robust password-based authenticated key exchange protocol for Cloud Computing in Mobile Cloud Computing (MCC), leveraging Chaotic Maps based on Discrete logarithmic and Diffie-Hellman problems. The protocol's strength is affirmed through a comprehensive security analysis, emphasizing specific findings regarding its resilience against threats like man-in-the-middle and replay attacks. Our work showcases notable performance gains, with a 60% reduction in execution time compared to existing schemes, although specific schemes are not detailed for brevity. Future research avenues include a more detailed comparison with existing schemes, exploration of advanced threat models, and consideration of additional performance metrics, providing a roadmap for further improving the protocol. To extend the impact of our work, we suggest delving into user-friendly aspects, evaluating usability and user experience, and addressing practical considerations for real-world deployment. By incorporating these extensions, our proposed protocol not only advances the state-of-the-art in both security and performance but also lays the groundwork for a more comprehensive and user-centric approach to secure communication in MCC environments.

REFERENCES

[1] Noor, T.H., Zeadally, S., Alfazi, A., Sheng, Q.Z. (2018). Mobile cloud computing: Challenges and future research directions. Journal of Network and Computer Applications, 115: 70-85. https://doi.org/10.1016/j.jnca.2018.04.018

[2] Fu, Z., Sun, X., Ji, S., Xie, G. (2016). Towards efficient content-aware search over encrypted outsourced data in cloud. In IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, Francisco, CA, USA, pp. 1-9. https://doi.org/10.1109/INFOCOM.2016.7524606

[3] Nawrocki, P., Reszelewski, W. (2017). Resource usage optimization in mobile cloud computing. Computer Communications, 99: 1-12. https://doi.org/10.1016/j.comcom.2016.12.009

[4] Thakare, S.V., Gore, D.V. (2014). Comparative study of CIA and revised-CIA algorithm. In 2014 Fourth International Conference on Communication Systems and Network Technologies, Bhopal, India, 713-718. https://doi.org/10.1109/CSNT.2014.150

[5] Mohammad, A.A.K., Mirza, A., Vemuru, S. (2016). Cluster based mutual authenticated key agreement based on Chaotic Maps for mobile ad hoc networks. Indian Journal of Science and Technology, 9(26): 1-11. https://doi.org/10.17485/ijst/2016/v9i26/95137

[6] Tsai, J.L. (2008). Efficient multi-server authentication scheme based on one-way hash function without verification table. Computers & Security, 27(3-4): 115-121. https://doi.org/10.1016/j.cose.2008.04.001

[7] Tsaur, W.J., Li, J.H., Lee, W.B. (2012). An efficient and secure multi-server authentication scheme with key agreement. Journal of Systems and Software, 85(4): 876-882. https://doi.org/10.1016/j.jss.2011.10.049

[8] Lee, C.C., Lou, D.C., Li, C.T., Hsu, C.W. (2014). An extended chaotic-maps-based protocol with key agreement for multiserver environments. Nonlinear Dynamics, 76: 853-866. https://doi.org/10.1007/s11071-013-1174-3

[9] Li, X., Niu, J., Kumari, S., Islam, S.H., Wu, F., Khan, M.K., Das, A.K. (2016). A novel Chaotic Maps-based user authentication and key agreement protocol for multi-server environments with provable security. Wireless Personal Communications, 89: 569-597. https://doi.org/10.1007/s11277-016-3293-x

[10] Irshad, A., Chaudhry, S.A., Xie, Q., Li, X., Farash, M.S., Kumari, S., Wu, F. (2018). An enhanced and provably secure chaotic map-based authenticated key agreement in multi-server architecture. Arabian Journal for Science and Engineering, 43: 811-828. https://doi.org/10.1007/s13369-017-2764-z

[11] Tsai, J.L., Lo, N.W. (2015). A privacy-aware authentication scheme for distributed mobile cloud computing services. IEEE Systems Journal, 9(3): 805-815. https://doi.org/10.1109/JSYST.2014.2322973

[12] He, D., Kumar, N., Khan, M. K., Wang, L., Shen, J. (2016). Efficient privacy-aware authentication scheme for mobile cloud computing services. IEEE Systems Journal, 12(2): 1621-1631. https://doi.org/10.1109/JSYST.2016.2633809

[13] Tan, Z. (2016). A privacy-preserving multi-server authenticated key-agreement scheme based on Chebyshev Chaotic Maps. Security and Communication Networks, 9(11): 1384-1397. https://doi.org/10.1002/sec.1424

[14] Irshad, A., Sher, M., Chaudhry, S.A., Xie, Q., Kumari, S., Wu, F. (2018). An improved and secure chaotic map based authenticated key agreement in multi-server architecture. Multimedia Tools and Applications, 77: 1167-1204. https://doi.org/10.1007/s11042-016-4236-y

[15] Jegadeesan, S., Azees, M., Kumar, P.M., Manogaran, G., Chilamkurti, N., Varatharajan, R., Hsu, C.H. (2019). An efficient anonymous mutual authentication technique for providing secure communication in mobile cloud computing for smart city applications. Sustainable Cities and Society, 49: 101522. https://doi.org/10.1016/j.scs.2019.101522

[16] Goh, E.J. (2003). Secure indexes for efficient searching on encrypted, compressed data. Technical Report 2003/216, Cryptology ePrint Archive, 2003.

http://eprint.iacr.org/2003/216

[17] Tan, S., Zhu, H., Wang, Y. (2009). Some notes on password authenticated key exchange based on RSA. In 2009 International Conference on Computational Intelligence and Security, Beijing, China, pp. 580-583. https://doi.org/10.1109/CIS.2009.225

[18] Slocum, Z. (2009). Your Google docs: Soon in search results?. http://news.cnet.com/8301-17939_109-10357137-2.html.

[19] Song, D.X., Wagner, D., Perrig, A. (2000). Practical techniques for searches on encrypted data. In Proceeding 2000 IEEE symposium on security and privacy. S&P 2000, Berkeley, CA, USA, pp. 44-55. https://doi.org/10.1109/SECPRI.2000.848445

[20] Chang, Y.C., Mitzenmacher, M. (2005). Privacy preserving keyword searches on remote encrypted data. In International conference on applied cryptography and network security, Springer, Berlin, Heidelberg, pp. 442-455. https://doi.org/10.1007/11496137_30

[21] Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R. (2006). Searchable symmetric encryption: improved definitions and efficient constructions. In Proceedings of the 13th ACM conference on Computer and communications security, pp. 79-88. https://doi.org/10.1145/1180405.1180417

[22] Gupta, P., Verma, D.K., Singh, A.K. (2018). Improving RSA algorithm using multi-threading model for outsourced data security in cloud storage. In 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, pp. 14-15. https://doi.org/10.1109/CONFLUENCE.2018.8442788

[23] Abdul, A.M., Jena, S., Balraju, M. (2020). Trust and Role Based Access Control to Mitigate the Malicious Activities of Authenticated Users in Mobile Cloud Computing. International Journal of Advanced Science and Technology, 29(5): 10627-10632. http://sersc.org/journals/index.php/IJAST/article/view/24185

[24] Zhu, H., Zhang, Y. (2017). An efficient Chaotic Maps-based deniable authentication group key agreement protocol. Wireless Personal Communications, 96: 217-229. https://doi.org/10.1007/s11277-017-4163-x

[25] Yin, A., Guo, Y., Song, Y., Qu, T., Fang, C. (2020). Two-round password-based authenticated key exchange from lattices. Wireless Communications and Mobile Computing, 2020, 8893628. https://doi.org/10.1155/2020/8893628

[26] Sun, H., Wen, Q., Zhang, H., Jin, Z. (2013). A novel remote user authentication and key agreement scheme for mobile client-server environment. Applied Mathematics & Information Sciences, 7(4): 1365. https://doi.org/10.12785amis/070414

[27] Tsai, J.L., Lo, N.W. (2015). A privacy-aware authentication scheme for distributed mobile cloud computing services. IEEE Systems Journal, 9(3): 805-815. https://doi.org/10.1109/JSYST.2014.2322973

[28] Jegadeesan, S., Azees, M., Kumar, P.M., Manogaran, G., Chilamkurti, N., Varatharajan, R., Hsu, C.H. (2019). An efficient anonymous mutual authentication technique for providing secure communication in mobile cloud computing for smart city applications. Sustainable Cities and Society, 49: 101522. https://doi.org/10.1016/j.scs.2019.101522