International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-12, Dec 2018 E-ISSN: 2347-2693

An Effective Trust-aware Authentication Framework for Cloud Computing Environment

SaboutNagaraju [1*], S.K.V. Jayakumar [2]

[1]Dept. of Computer Science, Pondicherry University Community College, Lawspet, Pondicherry-605 008, India
[2] Dept. of Computer Science, School of Engineering & Technology, Pondicherry University, Kalapet, Pondicherry-605 014, India

**Corresponding Author: saboutnagaraju1983@gmail.com, Tel.: +91-87787-01151*

Available online at: www.ijcseonline.org

Accepted: 06/Dec/2018, Published: 31/Dec/2018

Abstract— Although cloud computing has become one of the basic utility in ICT era with several benefits like rapid elasticity, resource pooling broad network access, and on-demand self-service, it introduces dozens of dirty security threats too. An effective authentication protocol is the basis, topmost prioritized and emergence one for the secure cloud communications. As a result, in this article an effective trust-aware authentication framework is proposed based on *n*-party multi-linear key pairing functions, trust and reputation aggregation functions and time-based dynamic nonce generation. In addition to formulating an effective authentication protocol, we have analyzed the mutual authentication and formal security strength by using cryptographic GNY belief logic which will prove proposed protocol not only meets intended mutual authentication, but also justifies the security strength against the impersonation and ephemeral secret leakage attacks.

Keywords— Mutual Authentication, Single Sign-On, Elliptic-Curve, Cloud Service Provider, Identity Provider, Trustee

I. INTRODUCTION

The recent development in Internet-of-Things, big data, mobile and social networks require cloud computing to provide economical data storage and high-speed computing capabilities. However, these imperatives are rapidly emerging as pillars for the smarter daily life and official works [1]. Although cloud computing has become one of the basic utility in ICT era with several benefits like rapid elasticity, resource pooling, broad network access, and on-demand self-service, it introduces dozens of dirty security threats too [2]. As per Cloud Security Alliance (CSA) research report 2018 [3], the top ten cloud specific security threats are data breaches, insufficient identity, credential and access management, web-based impersonations, insecure interfaces and APIs, system vulnerabilities, account hijacking, malicious insiders, advanced persistent threats, data loss, insufficient due diligence and denial of service. In [2], Bob Violino reported cloud authentication specific threats, among which the top five are data breaches, web-based impersonations, identity theft, account hijacking and malicious insiders. To preserve authenticity, authorization and key management properties in the cloud is given higher priority as these helps to protect the client's sensitive information from malicious users [4].

It is observed that each and every traditional web or mobile application usually itself authenticates and authorizes the users and stores all the credentials and authorization information required. In this traditional scenario, each user may have multiple accounts for different applications with the same or similar credentials. Traditional authentication and authorization methods have been working successfully well for a long time. However, for the security reasons it is better for the user to use different passwords for each application and need to change all of them regularly. It is a tough work to do for the user. Even today, in a few leading web applications user credentials are actually stored in unencrypted form [5, 6].

It would be easier for the user, if all the applications have a common user credentials database. Here, users can access all the applications using one set of login credentials. This type of authentication is called OpenID or single sign-on (SSO). OpenID is an authentication protocol developed by non-profit OpenID foundation with a centralized credentials database server [7, 8, 9, 10]. The centralized database can be managed by a trusted third party identity provider. Here, end users are allowed to access different web applications using same set of login credentials. The key advantage of this OpenID mechanism is to eliminate the need of webmasters to provide their own ad hoc login systems. The major problem in this protocol is a user is allowed to confirm his/her identity to a web application. This approach requires a trusted identity provider and it could become a bottleneck for the user authentication.

To overcome the problems of the OpenID mechanism, Chris Messina presented an OAuth 2.0 protocol [11, 12] not only for authentication but also for authorization of the users. In this protocol, a user is agrees to share his/her limited profile data from an OAuth identity provider. Here, a user can choose an OAuth identity provider like Microsoft, Google, Facebook, twitter etc. To choose or accept an Identity Providers (IdP) for the real-time applications should get recommendation from the OpenID Foundation. In [13, 14, 15, 16], security researcher reported that huge vulnerabilities are present in OAuth 2.0 libraries and OpenID approach and also demonstrated several security flaws in OAuth2.0 OpenSSL encryption process. OpenSSL has several security flaws due to that users access tokens and credentials are exposed to man in the middle attack and buffer overflow attacks. If OAuth2.0 is not configured correctly, it doesn't even look at the access token, it just checks the User-ID has come from the correct source. Hence, there is a chance of impersonation attack. Importantly, popular practical cloud-based authentication mechanisms [17-24] are designed based on OAuth2.0 protocol.

An existing OpenID based authentication schemes [27, 28] provides security and convenience for mobile users to access multiple mobile cloud computing services from multiple Cloud Service Providers (CSPs) using only a single private key. The authors have taken effort to supports mutual authentication, key exchange, user anonymity, and user untraceability in the cloud. In these investigations, user password and finger print details are never shared with the CSPs. However the mechanisms are insecure against the service provider impersonation attack and the adversary can able to extract the user identity. These schemes also not secure against the Ephemeral Secret Leakage (ESL) attack and malicious insiders.

The followings are the key problems identified from existing cloud-based authentication investigations [7-28]:

- In the cloud computing environment, very critical and barely explored issue that should be taken into consideration is impersonation attack that impersonates the cloud communication with the false responses. Due to insufficient identity credentials and nonintellectual access key management controls, impersonation attackers can steal user credentials and gain the control over outsourced data and applications.
- There is a chance of impersonation attack in the configurations of the OAuth2.0-based cloud authentications.
- Passwords are not enough and maintenance of numerous passwords increases security risks.
- To reduce number of complex operations involved in the authentication.
- For some financial/personal gain, dishonest cloud staff/ rogue system administrator may leak the user identities and access management details [29].
- Existing scheme is unrealistic in storing key credentials in the host device memory for identity verification.
- The development of an effective collaborative multi-factor authentication is critical.

In this article, we have presented a trust-aware authentication protocol to bring an appropriate solution for the above problems.

Rest of the paper is organized as follows, Section I presents literature reviews , Section II illustrates system-level model and assumptions, Section III provides system preliminaries, Section IV describes our investigation, section V discusses completeness of the proposed authentication protocol, Section VI reports the security and performance evaluation, Section VII concludes research work with future directions.

II. RELATED WORK

Developing an efficient, robust and more convenient mutual authentication mechanism for the distributed cloud computing environment is a challenging research work. This section presents the existing cloud-based authentication approaches that can meet stakeholder's requirements at some extend.

A. Risk-based Multi-factor Authentications

In [17], Merritt Maxim reported that Gigya Customer Identity Management (CIM) platform v6.5 is a market-leading secure identity and access management solution for public cloud SaaS applications that facilitates to the stakeholders to safeguard their cloud assets. The solution provides a Risk-based Multi-factor Authentication (RBMFA) using risk factors and One-time Password (OTP). Here, the first factor is risk parameters and the second factor is OTP via mobile or email. The risk parameters can be registered customer device and current location. When a consumer tries to access the cloud service accounts by using a new device, it authenticates the user via text SMS or voice call. Finally, this approach blacklists the consumer after a specified number of access attempts fail. This solution also facilitates user authentication through social accounts registration and logins or third-party plug-ins. The major strength of this solution is to provide password-less authentication and cross-network registration and login analytics. Compared to other cloud authentication controls, this approach provides the best execution, administration, analytics, partner ecosystem and reporting. This solution also has larger market presence and global presence of vendors. The major drawback of this approach is lacking in supporting TRUSTe security standards and certifications.

LoginRadius Identity and Access Management (LRIAM) mechanism [22] provides customizable identity solutions to securely access multitenant SaaS offering in the Microsoft Azure. LoginRadius platform supports a Risk-based Multi-factor Authentication (RBMFA). Here, the first factor is risk parameters and the second factor is OTP via mobile or email. The risk parameters can be registered customer device, network address and current location. This platform also facilitates the user authentication through social login, anonymous login, phone SSO login, federation SSO and two-factor Authentication (2FA). The major strength of this solution is to provide password-less authentication and federation SSO-based registration and logins. The major drawback of this approach is lacking in supporting TRUSTe security standards and certifications. The solutions also lack in providing appropriate partner ecosystem and secure customer data management.

Ping Identity and Access Management (PIAM) mechanism [21] provides market-leading authentication solutions to securely access public cloud SaaS applications. The solution platform easily integrates third-party identity providers' servers with the cloud service providers' servers. The solution provides a Risk-based Multi-factor Authentication (RBMFA) using risk factors and One-time Password (OTP). Here, the first factor is risk parameters and the second factor is OTP via mobile or email. The risk parameters can be registered customer device, network address and current location. This solution also facilitates the user authentication through social login linking or adaptive authentication policies. The major strength of this solution is to provide password-less authentication and cross-network registration and login analytics. Compared to other cloud authentication controls, this approach provides better execution, administration and partner ecosystem. The major drawback of this approach is lacking in supporting TRUSTe security standards and certifications. The solutions also lack in providing simplicity in the adaption customizations.

B. Single Sign-On Authentications

Janrain Identity Cloud [18] is another market-leading secure identity and access management solution for next generation cloud-based technologies such as IoT and big data networks. The solution ensures safe and seamless identity generation, establishment and management. It gives a set of options for the authentication based on user requirement, such as corporate login, mobile authentication, single sign-on, universal ID, social login, adaptive MFA authentication, etc,. The major strengths of the solution are to manage hierarchical groups to access each individual critical resources and uses Single Sign-On (SSO) to delivers one login across multiple applications and domains. Compared to other cloud authentication controls, this approach is better in overall performance, compliance management, threats and risks management and administration. This solution also has larger market presence, geographical presence of vendors and supports HIPAA, ISO and SOC2 security certifications and privacy compliance. The major drawback of this approach is lacking in protection of data breaches and transparent policy management.

Salesforce Identity and Access Management (SIAM) mechanism [19] is designed to securely authenticate various multitenant SaaS applications. In this approach, Salesfore SSO login system is implemented by using OAuth2 protocol strategy across multiple organizations. Salesfore SSO allows the consumers to authenticate into their multiple registered cloud services without having separate accounts for each service. Instead, a user can access all the applications using one set of login credentials. SIAM provides various administrative tools to monitor, report and maintain user authentication and authorization access tokens. Compared to other cloud authentication controls, this approach performs better in availability, scalability, security and privacy compliance management, customer data management, analytics and reporting. The major drawback of this approach is it has smallest CIAM installed base and performs immature user authentication. If OAuth2.0 is not configured correctly, it does not even look at the access token, it just checks the User-ID whether it comes from the correct source.

Azure Active Directory Business-to-Customer (AADB2C) solution [23] provides a seamless fully customizable identity and access management solution to securely access multitenant SaaS offering in the Microsoft Azure. Azure-AD also offers easy to use, consumer-centric, affordable and flexible CIAM solutions to the stakeholders to access multiple cloud applications by using single set of credentials. This solution delivers the user authentication through self-service password management, device registration, social and on-premise login, employees and business partners SSO login, two-factor Authentication (2FA) and federation SSO. Compared to other cloud authentication controls, Azure-AD performs better in scalability and performance. The major drawback of this approach is it has smallest CIAM installed base and performs immature authentication data analytics and reporting. The solutions also lack in providing appropriate partner ecosystem, content management solutions and geographical presence of vendors.

C. Multi-factor Authentications

ForgeRock Identity Platform (FRIP) [20] is a unified platform for secure user identity and access management services in the private cloud or on-premises applications. The solution provides a Time-based or HMAC-based Multi-factor Authentication (TBMFA or HBMFA) using user ID, password and One-time Password (OTP). Here, the first factor is user ID and password and the second factor is OTP via mobile or email or registered hard device. In this approach, OTP can be generated from the registered hardware device or apps and used in the user authentication. In another way OTP will be generated using hash functions or time interval specified. Specifically, the ForgeRock IAM services are suitable for data sharing and user consent. This approach preserves better privacy controls for data privacy protection compared to other mechanisms.This solution does not work for the user authentication in multitenant SaaS applications. The major drawback of this approach is it lacks in supporting TRUSTe security standards and certifications. This mechanism also performs immature user authentication in analytics and reporting.

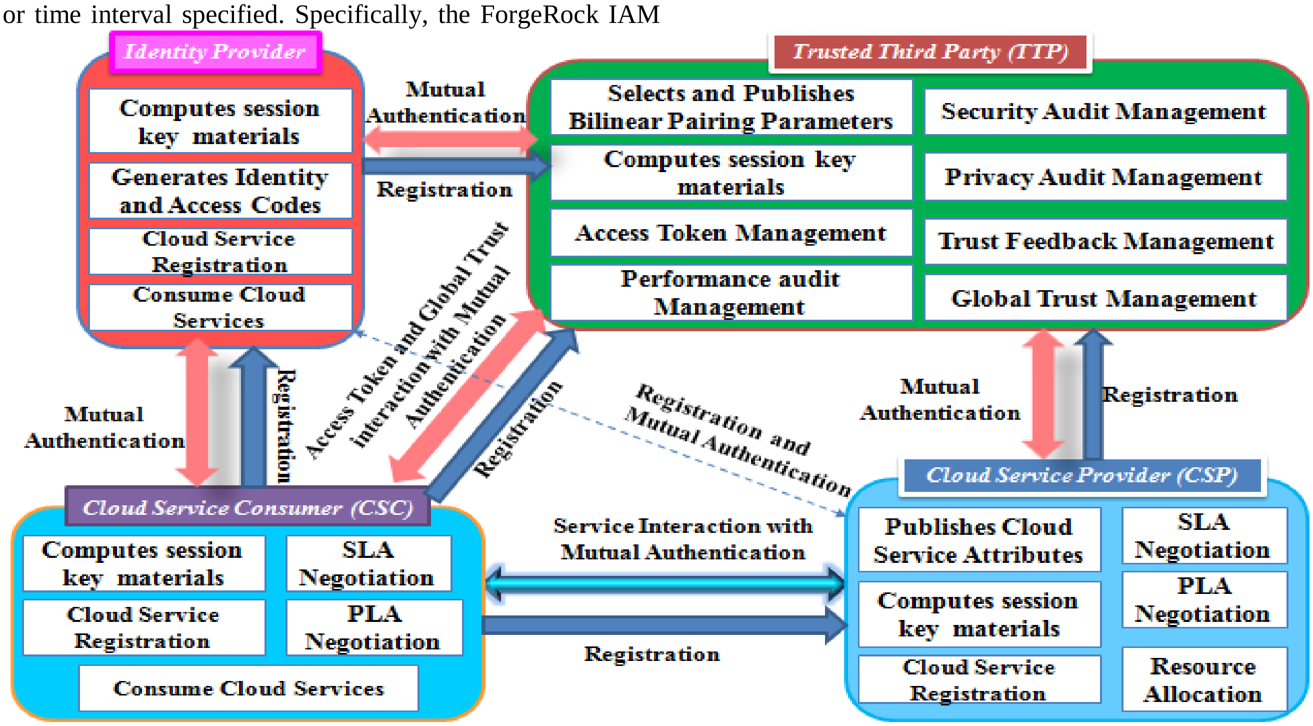

Figure.1. Typical Architecture of Proposed Scheme for cloud computing environment

In [24-26, 32-41] authors described various authentication frameworks for cloud computing environment to facilitate mutual authentication and secure key management for cloud users. Observing limitations, these schemes perform computational overhead and cannot secure against the Ephemeral Secret Leakage (ESL) attack. In [27], Jia-Lun Tsai et al. described a scheme to provide security and convenience for the mobile users to access multiple mobile cloud computing services from multiple service providers using only a single private key. The authors have taken effort to supports mutual authentication, key exchange, user anonymity, and user untraceability in the cloud. In this scheme user password and finger print details are never shared with CSPs and SCG. However the mechanism is insecure against the service provider impersonation attack and the adversary can able to extract the user identity. Debiao He et al [28] presented a privacy-aware authentication solution to address the impersonation problem exist in Jia-Lun Tsai et al. scheme. This scheme also not secure against the Ephemeral Secret Leakage (ESL) attack and malicious insiders.

III. SYSTEM-LEVEL FRAMEWORK AND ASSUMPTIONS

In this section a system-level framework is presented for distributed cloud computing environment which consists of cloud service providers, identity provider, distributed trustee and users as shown in Figure 1. The personal and sensitive information of a data owner or enterprise will be managed in the geographically distributed cloud data centers. The cloud service providers outsource cheap, flexible and on-demand storage space and computing capabilities to the data owner to make this information available any time to the legitimated users. Trustee is a set of distributed servers that are managed by an organization or board of eminent security researchers. It is a separation from the identity provider and cloud resource applications and will run on a separate trusted lockdowns security platforms. It collects and validates authentication codes and access tokens generated by the identity provider, if are valid, then user will be directed to access cloud application. Trustee protects authentication codes and access tokens and are never been processed in the service providers platforms. And also audits and records SLA and PLA parameters. Trustee services are distributed geographically with shared and highly secured databases. Identity provider is an authentication and authorization servers support to computes session key materials and generate identity and access codes for the user authentication. In our proposed framework, identity providers, legitimated users and CSPs must rely on distributed trustee. The notations and their meanings we used for describing our framework are listed in Table 1.

Table 1. List of Abbreviations

Notations	Meaning
U_i&CSP_j	User i and Cloud Service Provider *j*
E_{PB} (·)	A public-key encryption function
D_{PR}(·)	A decryption function's corresponding to E_{PB} (·)
e_K(·)	A symmetric encryption's function
d_K(·)	A symmetric decryption's function corresponding e_K(·)
ê	Exponential function
*UID**	The ID which U_i inputs in authentication phase
*PWD**	The password which U_i inputs in authentication phase
MAC_{Ui}*	The MAC address which U_i submits in authentication phase
UID	The ID which U_i inputs in registration phase
PWD	The password which U_i inputs in registration phase
MAC_{Ui}	The MAC address which U_i submits in registration phase
C_i	The cipher text sent by various communication entities.
T_{bp}	Bilinear pairing operation time
T_m	Multiplication operation time
T_d&T_c	Division operation time and Concatenation operation time
T_X&T_h	Ex-OR operation time and One-way Hash operation time
T_{dp}&T_{priv}	Trust on data processing and Trust on data privacy
R_{val}&GT_{val}	Reputation value and Global Trust value
α, β and γ	Importance given to service cost, trust and reputation values
T_{dt}&ST_{val}	Trust on data transmission and Service Trust value
K & K_i	Shared session key and Shared session key of communication entity *i*
ST & *DT*	Cloud Service Type and User Data Type
SS & *DS*	Cloud Storage Size and User Data Size
PS & *RS*	Cloud Processing Speed and User Requested Speed
SC & *SP*	Cloud Service Cost and User Service Pay
C_d	Cost Difference (i.e., $C_d=SC-SP$)
AT&RAT	Authentication Type and Requested Authentication Type
$C_{dminaccval}$	Minimum Acceptable Value of Cost Difference
$ST_{minaccval}$	Minimum Acceptable Value of Service Trust
$GT_{minaccval}$	Minimum Acceptable Value of Global Trust
SPID	The Service Provider ID which CSP_j inputs in the registration phase
*SPID**	The Service Provider ID which CSP_j inputs in the authentication phase
$h_i(.)$	i^{th} one-way hash function
‖&⊕	Concatenation operation and X-OR operation
$T(u, s)^t$	User *u* has the trust in service type *s* at current time *t*
salt	A random data that is used in generating a hashed password and also avoids the hash collisions.
+K & *-K*	Public keys and Private keys
ERN	Encrypted Random Number
FPR&FNR	False Positive Rate and False Negative Rate

IV. SYSTEM PRELIMINARIES

The *n*-party bilinear key pairing preliminaries we used in our proposed authentication protocol are described in this section. Let G_1, G_2, G_3 be three cyclic additively-written groups and let G_T be a cyclic multiplicative groups of an exponential base *g* with a large prime number order *p*.

Definition 4.1. Let a mapping $\hat{e}=G_1 \times G_2 \times G_3 \rightarrow G_T$ is a bilinear pairing that has characteristics as follow:

(1). Bilinearity:$\forall a, b, c \in F_q^*$, $\forall g \in (G_1, G_2, G_3)$, $\hat{e}(g^a, g^b, g^c)=\hat{e}(g, g, g)^{abc}$.

(2). Computability: Bilinear groups and bilinear mapping are computed efficiently.

(3). If $\hat{e}(g, g,g)=1$, then bilinear pairing preserves non-degeneracy property.

Definition 4.2. Let $\hat{e}$ be a bilinear pairing on (G_1 , G_2, G_3). The bilinear Elliptic Curve Diffie-Hellman key pairing for $\forall a,b, c \in F_q^*$, $\forall g \in (G_1, G_2, G_3)$ can be computed as $\hat{e}(g^a, g^b, g^c)=\hat{e}(g,g,g)^{abc}$.

The above definitions and properties are used in our authentication process for establishing and generating shared session keys among the users, identity providers, cloud service providers and trustee. In key generation process there are up-flow and down-flow stages. In up-flow stage, each entity computes intermediate secrete values and in the down-flow, intermediate results will be sent to the communication entity group to generate shared session keys. The

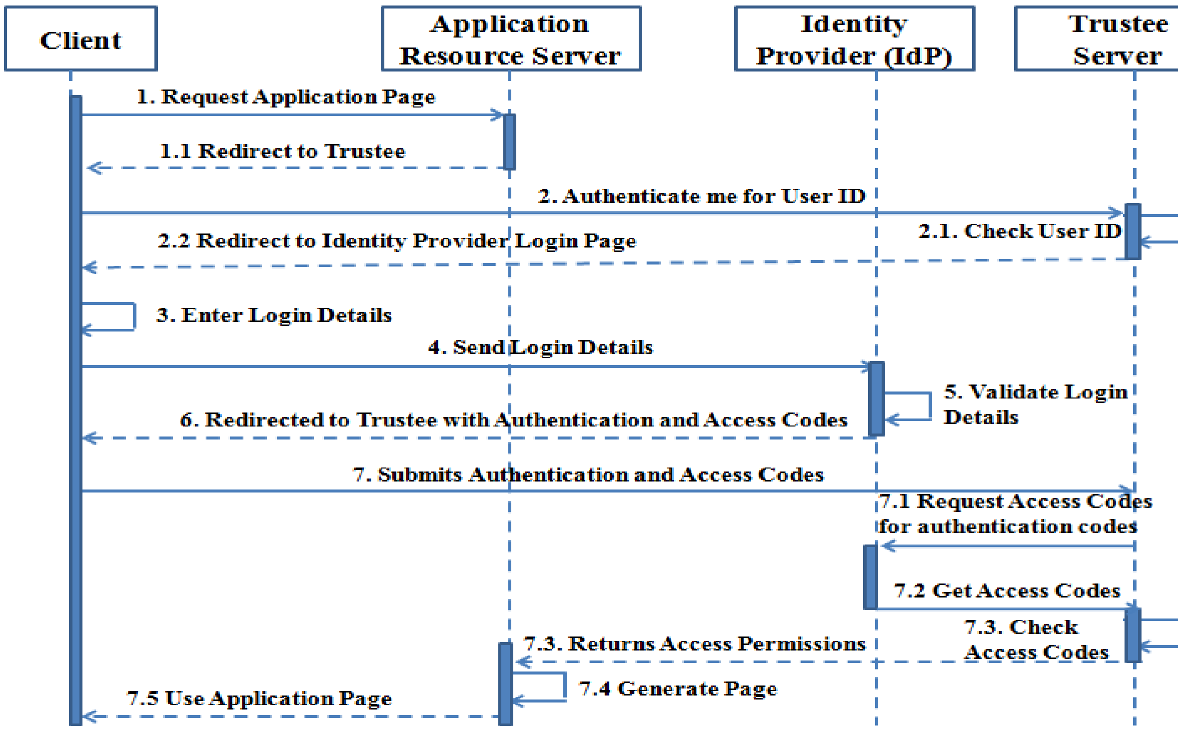

Figure.2. Control Flow of the Proposed Authentication Protocol

communication entities involved in authentication process are denoted as $E_1, E_2 \ldots E_n$. Trustee chooses an exponential base α and a large prime number p as an order and secretly shares these values to the authorized users and CSPs.

During up-flow, each communication entity E_i performs a single exponent and concatenates resultant value to the received intermediate values as given in equation (1) and then sends it to E_{i+1}.

$$E_i \xrightarrow{\alpha^{\prod(N_K|K\epsilon[i,j])}|j\epsilon[1,i]} E_{i+1} \quad (1)$$

E_{i+1} receives the up-flow as formulated in equation (2).

$$E_i \xrightarrow{(\alpha^{N_1}, \alpha^{N_1 N_2}, \ldots, \alpha^{N_1 N_2 \ldots N_i})} E_{i+1} \quad (2)$$

Upon receipt of the resultant flow, E_n computes the shared session key K as given in equation (3) by exponentiation of secrete value N_n chosen by E_n.

$$K = K_n = (\alpha^{N_1}, \alpha^{N_1 N_2}, \ldots, \alpha^{N_1 N_2 \ldots N_i})^{N_n} \quad (3)$$

The up-flow process ends and the down-flow process starts when $E_i = E_n$. Once the shared session key K_n is computed, E_n starts the down-flow with n-1 intermediate values as formulated in equation (4)

$$(\alpha^{N_1 N_n}, \alpha^{N_1 N_2 N_n}, \ldots, \alpha^{N_1 N_2 \ldots N_{n-2} N_n}) \quad (4)$$

Upon receipt of n-1 intermediate values, each entity E_i computes the shared session key as given in equation (5)

$$K = K_i = (\alpha^{N_1 N_2 \ldots N_{i-1} N_{i+1} \ldots N_n})^{N_i} \quad (5)$$

The down-flow ends when $E_i = E_1$.

V. TRUSTED AUTHENTICATION PROTOCOL

In this section we describe a trust-aware mutual authentication protocol. In this protocol, authentication parameters' matching will be performed in the identity provider servers. The authentication codes and access tokens generated by the identity provider will be validated in the distributed trustee servers. In our approach, user authentication credentials never shared with the cloud service providers. The control flow of the proposed authentication protocol is represented in Figure.2. This approach helps the users to protect identity and access management tokens from the malicious insiders and unauthorized external adversaries.

The protocol has three phases as follow.

Initialization phase, First, *trustee* chooses a random number as private key (Pr_{tt}) and computes $Pb_{tt} = h_1(Pr_{tt})$ as its corresponding public key, where h_1 is a one-way hashing function. Next *trustee* selects various bilinear pairing function parameters (p, a, b, G, h_2 to h_5 and n). Finally, *trustee* publishes Pb_{tt} and (p, a, b, G, n, h_2 to h_5) as public parameters. Likewise, identity provider (*IdP*) chooses a random number as private key (Pr_{IdP}) and computes $Pb_{IdP} = h_2(Pr_{IdP})$ as its corresponding public key, where h_2 is a one-way hashing function and publishes Pb_{IdP} as a public parameter. Similarly, *CSP* chooses a random number as private key (Pr_{CSP}) and computes $Pb_{CSP} = h_3(Pr_{CSP})$ as its corresponding public key, where h_3 is a one-way hashing function and publishes Pb_{CSP} and its service attributes.

Registration phase, Each User (U_i) filters CSPs based on service attributes like {*STЭDT, SS≥DS, PS≥RS, SC≤SP*} and then sends a request to *trustee* to provide trust and reputation values of desired *CSP*. User (U_i) sends his/her chosen CSP_{IDj}, user-id (ID_{Ui}), password (pwd_{Ui}) and device MAC address (MAC_{Ui}) to *IdP* for registration. Where, each user selects desired cloud service provider (CSP_{IDj}) based on the global trust evaluation algorithm which is described in [26]. Identity provider computes $h_3(PWD+salt)=HPWD$, $h_4(\delta_{RN}(MAC)) = h_4(RN \oplus MAC) = HMAC$ and $e_{MACUi}(RN) = ERN$, where $h_3(.)$ and $h_4(.)$ are the one-way hashing functions, $e_{MACUi}(.)$ is the symmetric encryption function using MAC_{Ui} and stores these values in distributed and highly secured databases. *IdP* sends ID_{Ui} and mutual operation on *nonce* to U_i and *trustee* through secure channel.

Assumption: Similarly, trustee and cloud service provider registers with IdP.

The authentication phase performs the following steps to validate remote user (U_i) login credentials.

1) User U_i inputs ID_{Ui}*, chooses a random secrete number x $(x<p)$ and nonce n_1 and then computes intermediate secrete as $X_x = g^x$ mod p. U_i performs public key encryption on concatenation of ID_{Ui}*, X_x and n_1 and computes cipher text as $C1=E_{PB_{TT}}(ID_{Ui}*||CSP_{IDj}*||n_1)||X_x$ and then sends C_1 as service request to $Trustee_k$.
2) $Trustee_k$ obtains U_i message details such as ID_{Ui}*, CSP_{IDj}* and n_1 by decrypting C_1 using private key PR_{TT}. If $UID*==UID$, then $Trustee_k$ selects a random secrete number y ($y<p$) and calculates $X_y = g^y \bmod p$, $X_{xy}=\hat{e}(X_x, X_y) \bmod p$, $n_2 = n_1>>1 \bmod n$ and then derives $C_2= E_{pbIdP}(ID_{Ui}*||ID_{TTk}*||n_2)||X_x|| X_y||X_{xy}$ using identity provider public key PB_{IDP}. $Trustee_k$ sends C_2 to the identity provider. If ID_{Ui}* or CSP_{IDj}* is not found or invalid, then the user request will be rejected.
3) Identity provider obtains $Trustee_k$ message details such as ID_{Ui}*, ID_{TTk}*, n_2, X_x, X_y, X_{xy} and n_2 by decrypting C_2 using private key PR_{IdP}. If $ID_{Ui}^*==ID_{Ui}$ && $ID_{TTi}*==ID_{TTi}$ &&valid? then chooses a secrete z $(z<p)$ and computes $X_z = G^z \bmod p$, session key $k= X_{xyz} = e(X_x, X_y)^z \bmod p$, $X_{xz} = e(X_x, X_z) \bmod p$, $X_{yz} = e(X_y, X_z) \bmod p$ and also performs the mutual operation on $n_3 = n_2>>1 \bmod n$. Finally, computes $C_3=E_k(ERN||n_3)||X_x||X_y|| X_z||X_{xz}||X_{yz}$ and *IdP* sends C_3 to the user. If *UID**or *SPID** is not found with trustee, then theauthentication request will be rejected.
4) FromC_3, U_i Computes session key $k= X_{xyz} = e(X_y, X_z)^x \bmod p$, obtains $ERN||n_3$ and then checks for mutual authentication value *i.e.*,$n_3= = n_1 >>2 \bmod n$, if it matches, then user is allowed to enter pwd_{Ui} and MAC_{Ui} and then obtains *RN* by decryption of message as $d_{pwdUi}(e(pwd_{Ui}(RN))=RN$. Finally, U_i computes $h_4(RN \oplus MAC_{Ui}*)=HMAC*$ and $C_4= E_k(HMAC*||n_4)$ and then sends C_4 to *IdP*. If $n_3 \neq n_1 >>1 \bmod n$, then the authentication process will be terminated.
5) From C_4, IdP_j obtains *HMAC** from $D_k(E_k(HMAC*||n_4))$ and checks for $HMAC*==HMAC$ && $n_4== n_3>>1 \bmod n$? if matches, then computes authentication and access token $E_k(Token_{TT}||n_5)$, where $Token_{TT}= E_{pbTT}(ID_{Ui}^*||NA_{Ui}||OTP||n_5)$ and $n_5 = n_4>>1 \bmod n$ and computes $C_5= E_k(Token_{TT}||n_5)$ and then sends C_5 to U_i.
   If $n_4 \neq n_3>>1$, then the authentication process will be terminated.
6) From C_5, U_i obtains $Token_{TT}||n_5$ by using secrete key k and then checks for $n_5= = n_4 >>1 \bmod n$?, if matches then user is allowed to enter *OTP* and forms $C_6= E_k(Token_{TT}||NA_{Ui}*||OTP||n_6)||X_x$, and then sends C_6 to $Trustee_k$. If $n_5 \neq n_4>>1$, then the authentication process will be terminated.
7) $Trustee_k$ computes $k = e(X_{x,z})^y \bmod p$ and obtains $Token_{TT}||NA_{Ui}*||OTP*||n_6$ using k and then obtains $ID_{Ui}^*||NA_{Ui}||OTP||n_5$ using Pr_{TT} and then checks for $ID_{Ui}^*==ID_{Ui}$&& $NA_{Ui}*== NA_{Ui}$&&$OTP*==OTP$&& $n_6==n_5>>1 \bmod n$? if matches, then redirects to *CSP* applications with $C_7= E_k(n_7)$ and then sends C_7 to U_i. If $n_6 \neq n_5>>1$, then the authentication process will be terminated.
8) U_i checks for mutual authentication value as $n_7= = n_6>>1 \bmod n$, if it matches, then user is allowed to access the cloud services. Otherwise, the request will be rejected.

The proposed mutual authentication protocol is described in Algorithm 1.

Algorithm 1: Authentication phase

Input: *User-ID*, *password*, *MAC* address and random *nonce*.
Output: Accept or Reject remote user.

1) U_i Inputs ID_{Ui}* and selects x $(x<p)$ and n_1
   Computes $X_x=g^x \bmod p$
   $C_1=E_{PbTT}(ID_{Ui}*||CSP_{IDj}*|| n_1)|| X_x$
   $U_i \xrightarrow{C_1} Trustee_k$
2) $Trustee_k$ $D_{PrTT}(E_{PbTT}(ID_{Ui}*||CSP_{IDj}*||n_1)= (ID_{Ui}*||CSP_{IDj}*||n_1)$
   if $ID_{Ui}*==ID_{Ui}$ and $CSP_{IDj}*==CSP_{IDj}$ and are *valid* then chooses a secrete
   number y (y<p) and computes $X_y = g^y \bmod p$, $X_{xy} = \hat{e}(X_x, X_y) \bmod p$, $n_2 = n_1>>1 \bmod n$
   $C_2= E_{pbIdP}(ID_{Ui}*||ID_{TTk}*||n_2)||X_x|| X_y||X_{xy}$
   $Trustee_k \xrightarrow{C_2} IdP_j$
   If ID_{Ui}* or CSP_{IDj}* is not found or invalid, then user request will be rejected
3) IdP_j decrypts C_2 as $D_{PrIdP}(E_{pbIdP}(ID_{Ui}*||ID_{TTk}*||n_2))= (ID_{Ui}*||ID_{TTk}*||n_2)$ and obtains ID_{Ui}*, ID_{TTk}*, n_2, X_x, X_y, X_{xy}
   if $ID_{Ui}^*==ID_{Ui}$ && $ID_{TTi}*==ID_{TTi}$ &&valid?, then chooses a secrete z $(z<p)$ and computes $X_z = G^z \bmod p$, session key $k= X_{xyz} = e(X_x, X_y)^z \bmod p$, $X_{xz} = e(X_x, X_z) \bmod p$, $X_{yz} = e(X_y, X_z) \bmod p$ and also performs mutual operation as $n_3 = n_2>>1 \bmod n$ and computes $C_3=E_k(ERN||n_3)||X_x||X_y|| X_z||X_{xz}||X_{yz}$

$IdP \xrightarrow{C_3} U_i$

Otherwise, the authentication request will be rejected.

4) U_i Computes session key $k= X_{xyz} =e(X_y, X_z)^x \bmod p$, obtains $ERN||n_3$ and then checks for mutual authentication value *i.e.*,$n_3 = = n_2 >>1 \bmod n$, if it matches, then user is allowed to enter pwd_{Ui} *and* MAC_{Ui} and then obtains RN by $d_{pwdUi}(e(pwd_{Ui}(RN))=RN$, finally computes $h_4(RN \oplus MAC_{Ui}*)=HMAC*$ and forms $C_4= E_k (HMAC*||n_4)$.

$$U_i \xrightarrow{C_4} IdP_j$$

If $n_3 \neq n_1 >>1 \bmod n$, then the authentication process will be terminated.

5) From C_4, IdP_j obtains $HMAC*$ from $D_k(E_k (HMAC*||n_4))$ and checks for $HMAC*==HMAC \,\&\&\, n_4== n_3>>1 \bmod n$? if matches, then computes authentication and access token $E_k(Token_{TT}||n_5)$, *where* $Token_{TT}= E_{pbTT} (ID_{Ui}^*||NA_{Ui}||OTP\ ||n_5)$ *and* $n_{5=} n_4>>1 \bmod n$ and computes $C_5= E_k (Token_{TT}||n_5)$

$Trustee \xrightarrow{C_5} U_i$ and redirects to the $Trustee_k$ server

If $n_4 \neq n_3>>1$, then the authentication process will be terminated.

6) From C_5, U_i obtains $Token_{TT}||n_5$ by using secrete key k and then checks for $n_5= = n_4 >>1 \bmod n$?, if matches then user is allowed to enter OTP and forms $C_6= E_k (Token_{TT}||NA_{Ui}*||OTP||n_6)||X_x$,.

$U_i \xrightarrow{C_6} Trustee_k$

Otherwise, the authentication request will be rejected.

7) $Trustee_k$ computes $k =e(X_{x,z})^y \bmod p$ and obtains $Token_{TT}|| NA_{Ui}*||OTP*||n_6$ using k and then obtains $ID_{Ui}^*||NA_{Ui}|| OTP||n_5$ using Pr_{TT} and then checks for $ID_{Ui}^*==ID_{Ui}\&\& NA_{Ui}*== NA_{Ui}\&\&OTP*==OTP\&\& n_6==n_5>>1 \bmod n$? if matches, then redirects to CSP applications with $C_7= E_k (n_7)$

$Trustee_k \xrightarrow{C_7} U_i$

Otherwise, the authentication request will be rejected.

8) U_i checks for mutual authentication value as $n_7= = n_6>>1 \bmod n$, if it matches, then user is allowed to access the cloud services. Otherwise, the request will be rejected.

VI. COMPLETENESS OF THE PROPOSED PROTOCOL

This section formally analyses the mutual authentication and security strength of the proposed protocol using standard GNY cryptographic logic. The analysis proved that the proposed protocol not only meets intended mutual authentication functionality, but also ensures the security strength against the service provider impersonation and other replay attacks. We used cryptographic GNY[30] belief logic to formally analyze the working nature of our trusted authentication mechanism and to verify whether our mechanism meets its goals. GNY belief logic is the substantial extension of BAN logic. First, we present the basic terminologies and statements, protocol transformation, goals and assumption list we used. Next, we describe the logical postulates adoption.

1) *Basic Terminologies and Statements*

Let CP_i be the credential parameter message and the following basic terminologies are introduced on CP_i:

- $h(CP_i)$: hash operation on CP_i.
- $\{CP_i\}_{+K}$, $\{CP_i\}_{-K}$: CP_i is encrypted with $+K$ and decrypted with $-K$.
- $\{CP_i\}_K$, $\{CP_i\}^{-1}_K$: CP_i is encrypted and decrypted with secrete key K.

Statements: Let E_i and E_j be two communication entities and the following statements are formed on E_i and E_j.

1) $E_i \lhd E_j$: E_i holds E_j
2) $E_i \ni CP_i$: E_i possesses credential parameter message CP_i
3) $E_i|\sim CP_i$: E_i once conveyed CP_i
4) $E_i|\equiv\#(CP_i)$:E_i believes that CP_i is fresh
5) $E_i\equiv\phi(CP_i)$:E_i believes that CP_i is recognizable
6) $E_i|\equiv E_i \overset{S}{\leftrightarrow} E_j$: E_i believes that S is a suitable secrete for E_i and E_j
7) $E_i|\equiv \overset{+K}{\rightarrow} E_j$: E_i believes that public key $+K$ is suitable for E_j
8) $E_i=>X$: E_i has jurisdiction over X
9) $E_i \lhd *X$: E_i is told that he/she didn't convey X previously in the current session.

2) *Protocol Transformation*

Our proposed authentication protocol is mapped into the form of $E_i \rightarrow E_j$:CP_i

1) $U_i \rightarrow Trustee_k$:$\{\{ID_{Ui}*||CSP_{IDj}*|| n_1\}_{+K}|| X_x\}$
2) $Trustee_k \rightarrow IdP_j$:$\{\{ID_{Ui}*||ID_{TTk}*||n_2\}_{+K} ||X_x|| X_y||X_{xy}\}$
3) $IdP_j \rightarrow U_i$:$\{\{ERN ||n_3\}_K||X_x||X_y|| X_z||X_{xz}||X_{yz}\}$
4) $U_i \rightarrow IdP_j$:$\{\{HMAC*||n_4\}_K\}$
5) $IdP_j \rightarrow U_i$:$\{\{Token_{TT}||n_5\}_K\}$
6) $U_i \rightarrow Trustee_k$:$\{\{Token_{TT}||NA_{Ui}*||OTP||n_6\}_{+K}||X_{x,z}\}$
7) $Trustee_k \rightarrow U_i$:$\{\{n_7\}_{+K}\}$

Parsing of the authentication protocol into $E_i |\sim CP_i$ and $E_i \lhd *X$ is given below.

1) $Trustee_k \lhd *\{*ID_{Ui}*||*CSP_{IDj}*|| *n_1\}_{+K}$ -> $U_i|\equiv U_i \overset{+K}{\leftrightarrow} Trustee_k$
2) $IdP_j \lhd *\{*\{*ID_{Ui}*||*ID_{TTk}*||*n_2\}_{+K} ||*X_x|| *X_y||*X_{xy}\}$-> $Trustee_k |\equiv Trustee_k \overset{+K}{\leftrightarrow} IdP_j$
3) $U_i \lhd *\{*\{*ERN||*n_3\}_K||*X_x||*X_y|| *X_z||*X_{xz}||*X_{yz}\}$-> $IdP_j |\equiv IdP_j \overset{K}{\leftrightarrow} U_i$
4) $IdP_j \lhd *\{*HMAC*||*n_4\}_K$ -> $U_i |\equiv U_i \overset{+K}{\leftrightarrow} IdP_j$
5) $U_i \lhd *\{*\{*Token_{TT}||*n_5\}_K\}$-> $IdP_j |\equiv IdP_j \overset{+K}{\leftrightarrow} U_i$
6) $Trustee_k \lhd *\{*\{*Token_{TT}||*NA_{Ui}*||*OTP||*n_6\}_{+K} ||*X_{x,z}\}$ -> $U_i|\equiv U_i \overset{+K}{\leftrightarrow} Trustee_k$
7) $U_i \lhd *\{*n_7\}_K$ -> $Trustee_k |\equiv Trustee_k \overset{+K}{\leftrightarrow} U_i$

A. Goals

The followings are the goals which describe the basic functionalities of the proposed protocol.

1) Authentication on message content

In the proposed protocol, $Trustee_k$ believes that user login request contents are recognizable and valid

$$Trustee_k \mid\equiv \phi \{\{ID_{Ui}^*||CSP_{IDj}^*|| n_1\}_{+K}|| X_x\}.$$

In the second flow, IdP_j believes that $Trustee_k$ message contents are recognizable and valid

$$IdP_j \mid\equiv \phi \{\{ID_{Ui}^*||ID_{TTk}^*||n_2\}_{+K} ||X_x|| X_y||X_{xy}\}.$$

In the third flow, U_i believes that IdP_j message contents are recognizable and valid

$$U_i\mid\equiv \phi\{\{ERN ||n_3\}_K||X_x||X_y|| X_z||X_{xz}||X_{yz}\}.$$

In fourth flow, IdP_j believes that U_i response contents are recognizable and valid

$$IdP_j \mid\equiv \phi\{\{HMAC^*||n_4\}_K\}.$$

In fifth flow, U_i believes that IdP_j reply message contents are recognizable and valid

$$U_i\mid\equiv \phi\{\{Token_{TT}||n_5\}_K\}.$$

In sixth flow, $Trustee_k$ believes that U_i message contents are recognizable and valid

$$Trustee_k \mid\equiv \phi\{\{Token_{TT}||NA_{Ui}^*||OTP||n_6\}_{+K}||X_{x,z}\}.$$

In seventh flow, U_i believes that $Trustee_k$ reply message contents are recognizable and valid

$$U_i\mid\equiv \phi\{\{n_7\}_{+K}\}.$$

2) Authentication on message origin

From the login request, $Trustee_k$ believes that U_i is originated the following message:

$$Trustee_k \mid\equiv U_i\mid\sim\{\{ID_{Ui}^*||CSP_{IDj}^*|| n_1\}_{+K}|| X_x\}.$$

In the second flow, IdP_j believes that $Trustee_k$ redirected the user with following message:

$$IdP_j \mid\equiv Trustee_k \mid\sim\{\{ID_{Ui}^*||ID_{TTk}^*||n_2\}_{+K} ||X_x|| X_y||X_{xy}\}.$$

In the third flow, U_i believes IdP_j is replied

$$U_i\mid \equiv IdP_j \mid\sim\{\{ERN ||n_3\}_K||X_x||X_y|| X_z||X_{xz}||X_{yz} \}.$$

In the fourth flow, IdP_j believes U_i is replied

$$IdP_j \mid \equiv U_i\mid\sim\{\{HMAC^*||n_4\}_K\}.$$

In the fifth flow, U_i believes and validates IdP_j response

$$U_i \mid\equiv IdP_j \mid\sim\{\{Token_{TT}||n_5\}_K\}.$$

In the sixth flow, $Trustee_k$ believes that U_i is replied

$$Trustee_k \mid \equiv U_i\mid\sim\{\{Token_{TT}||NA_{Ui}^*||OTP||n_6\}_{+K}||X_{x,z}\}.$$

In the seventh flow, U_i believes and validates $Trustee_k$ response

$$U_i \mid\equiv Trustee_k \mid\sim\{\{Token_{TT}||n_5\}_K\}.$$

3) Mutual Identity Verification

From the first flow, $Trustee_k$ believes and verifies ID_{Ui}^* and CSP_{IDj}^*, if identities are valid and matched then $Trustee_k$ sends ID_{Ui}^*, ID_{TTk}^* and n_2 to IdP_j, otherwise user request will be terminated

$$Trustee_k \mid\equiv U_i \ni (ID_{Ui}^*).$$

From the second flow, IdP_j believes and verifies ID_{Ui}^* and ID_{TTk}^*, if identities are valid and found, then IdP_j sends the intermediate secretes and encrypted random number (*ERN*) and n_3 to U_i, otherwise authentication request will be terminated

$$IdP_j \mid\equiv U_i\ni(ID_{Ui}^*) \;\&\&\; Trustee_k\ni(ID_{TTk}^*).$$

From the third flow, U_i verifies *ERN* and n_3, if $n_3 = = n_2 >> 1 \bmod n$, then user believes that the response received is genuine, otherwise authentication process will be stopped

$$U_i\mid\equiv Trustee_k, CSP_j\ni(n_3).$$

From the fourth flow, IdP_j verifies U_i incremented *nonce* value and $HMAC^*$, if $n_4 \neq n_3 >> 1 \bmod n$ and MAC address is matched, then IdP_j believes that the response received from U_i is genuine, otherwise the authentication process will be terminated

$$IdP_j\mid\equiv U_i\ni(HMAC^*, n_4).$$

From the fifth flow, U_i verifies IdP_j incremented nonce n_5, if $n_5 \neq n_4 >> 1 \bmod n$, then U_i believes that the response received from IdP_j is genuine and user is allowed to enter *OTP* and forms $C_6 = E_k\ (Token_{TT}||NA_{Ui}^*||OTP||n_6)||X_x$; otherwise the authentication process will be terminated

$$U_i \mid\equiv IdP_j\ni(Token_{TT}, n_5).$$

From sixth flow, $Trustee_k$ believes that U_i message contents are recognizable and valid

$$Trustee_k \mid\equiv U_i\ni(Token_{TT}, NA_{Ui}^*, OTP \text{ and } n_6).$$

In seventh flow, U_i believes that $Trustee_k$ reply message contents are recognizable and valid, then U_i checks for mutual authentication value as $n_7 = = n_6 >> 1 \bmod n$, if it matches, then user is allowed to access the cloud services. Otherwise, the request will be rejected

$$U_i\mid\equiv Trustee_k(Token_{TT}, NA_{Ui}^*, OTP \text{ and } n_6)$$

B. Session Key Material Establishment

U_i, IdP_j and $Trustee_k$ believes each other that X_x, X_y and X_z are their intermediate secrete values for generating shared session key

$$U_i\mid\equiv Trustee_k \mid\equiv IdP_j \mid\equiv \{U_i, IdP_j, Trustee_k\}\ni\{ X_x, X_y, X_z \}.$$

U_i, IdP_j and $Trustee_k$ believes that K is a shared one-time secrete key for the current session

$$U_i\mid\equiv IdP_j\mid\equiv Trustee_k\mid\equiv\{U_i \overset{K}{\leftrightarrow} IdP_j,\quad IdP_j \overset{K}{\leftrightarrow} Trustee_k,\quad U_i \overset{K}{\leftrightarrow} Trustee_k\}.$$

C. Assumption List

We consider the following assumptions in our authentication protocol.

- $Trustee_k$ chooses a random values as private key $-K$, computes corresponding public key $+K$ and prepares a one-time intermediate secrete value X_z for generating shared session key

$$Trustee_k \ni -K,\ Trustee_k \ni +K,\ Trustee_k \ni X_z.$$

- $Trustee_k$ publishes a public key $+K$ for the users and identity providers to encrypt their communication messages and also believes that $+K$ is suitable for IdP_j and U_i.

$$Trustee_k\mid\equiv \overset{+K}{\longrightarrow}\{ IdP_j, U_i\}.$$

- IdP_j chooses a random value as private key–K, computes corresponding public key $+K$ and prepares a one-time intermediate secrete value X_y for generating shared session key
$$IdP_j \ni -K,\ IdP_j \ni +K,\ IdP_j \ni X_y.$$
- IdP_j publishes a public key $+K$ for the users to encrypt their communication parameters and believes that $+K$ is suitable for U_i.
$$IdP_j \mid\equiv \xrightarrow{+K} \{U_i\}.$$
- U_i chooses a random value 'x' and prepares one-time intermediate secrete X_x. U_i believes that X_x is fresh and it will be used by $Trustee_k$ and IdP_j to compute shared secrete keys
$$U_i \ni X_x,\ U_i \equiv \#(X_x).$$
- IdP_j chooses a random value 'y' and prepares one-time intermediate secrete X_y. IdP_j believes that X_y is fresh and it will be used by $Trustee_k$ and U_i to compute shared secrete keys
$$IdP_j \ni X_y,\ IdP_j \equiv \#(X_y).$$
- $Trustee_k$ chooses a random value 'z' and prepares one-time intermediate secrete value X_z. $Trustee_k$ believes that X_z is fresh and it will be used by IdP_j and U_i to calculate shared secrete keys
$$Trustee_k \ni X_z,\ Trustee_k \equiv \#(X_z).$$

VII. PERFORMANCE EVALUATION

In this section we establish the testbed simulation using Microsoft Azure Compute and Storage Emulator. Using simulation platform we determine an effectiveness of the proposed protocol in terms of number of cryptographic operations are required, Resistance to various possible attacks, communication and computation costs. In first subsection, we present security comparisons. Next, we analyse the computational efficiency of our scheme with an existing schemes.

***Setup*:** We have implemented our proposed investigation on a computer which has windows 7 operating system with 4GB RAM and 2.0GHz Intel Core i7 processor. C#.NET framework was installed on this computer which contains Visual Studio community 2013 as a frontend, SQL Server 2012 R2 SP1 as a backend and a Windows Azure Emulator as software platform.

A. Security Comparisons

In this subsection, first we compare proposed authentication protocol with the existing mechanisms [17]-[22], [27]-[28] in terms of mutual authentication, resistant to various reply and impersonation attacks, and trust and reputations management attacks. As presented in Table II, the existing mechanisms [17]-[22] and [27]-[28] are effortless to protect trust and reputation management attacks such as white-wash attack, collusion attack, bad mouth attack and good mouthing attack.

The mechanisms described in [17]-[22] are not suitable for collaborative cloud service providers. Existing mechanisms presented in [20]-[27] are not resistance to reply and impersonation attacks. The schemes presented in [17] and [20]-[22] are unable support mutual authentication.However, the mechanisms described in [27]-[28] are unable support provision of user anonymity. Therefore, our investigation meets all the design goals and is immune to various reply and impersonation attacks.

Table 2.Comparisons with Existing Mechanisms

	CP_1	CP_2	CP_3	CP_4	CP_5	CP_6	CP_7	CP_8	CP_9
Merritt Max. [17]	No	No	Yes	Yes	Yes	No	No	No	No
Janrain. [18]	Yes	No	Yes	Yes	Yes	No	No	No	No
SIAM .[19]	Yes	No	Yes	Yes	Yes	No	No	No	No
David G.[20]	No	No	No	No	Yes	No	No	No	No
John T.[21]	No	No	No	No	Yes	No	No	No	No
LoginRadius [22]	No	No	No	No	Yes	No	No	No	No
Jia-Lun T et al.[27]	Yes	No	No	No	No	No	No	No	No
Debiao H et al. [28]	Yes	No	Yes	No	No	No	No	No	No
Our's	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

CP_1: Mutual authentication
CP_2: Collaborative Cloud Service Providers
CP_3: Resistance to replay attack
CP_4: Resistance to impersonation attack
CP5: Provision of user anonymity
CP6: White-washing attack
CP7: Collusion attack
CP8: Bad mouthing attack
CP9: Good mouthing attack

Next, we compare the computation costs of our authentication protocol with an existing scheme [28]. Let T_{bp} be the bilinear pairing operation time, T_h is one-way hash operation time, T_c and T_x are concatenation and Exclusive-OR operation times and, T_i and T_m are inverse and additive multiplication operation times respectively. The comparison of the computation cost of our scheme with an existing mechanism [28] is listed in Table III. In general, concatenation and bitwise Exclusive-OR operations are much faster and will consume constant timings, so that these two operations time can be neglected in calculating computation cost. Therefore, for registration process, our scheme requires two hash and one Exclusive-OR operations (i.e., $2T_h + T_x$). On the other hand, Debiao He et al. [28] scheme consumes two bilinear pairing, two hash, two multiplication and two inverse operations (i.e.,$2T_{bp}+2T_h+2T_m+2T_i$). For authentication process, proposed protocol consumes $3T_{bp}+T_h$ and Debiao He et al. scheme requires $7T_{bp}+6T_h+4T_m+2T_i$. So, the total computation cost of our authentication mechanism is O($3T_{bp}+2T_h+3T_m+19T_c+2T_x$) and Jia-Lun T et al. scheme consumes O($7T_{bp}+6T_h+4T_m+2T_i+5T_c+2T_x$).

Table 3. Computation Cost Comparisons with Existing Scheme

Phase	Party	Existing Scheme [28] Time Complexity	Our scheme Time Complexity
Registration	User	$T_{bp}+T_h+T_m+T_i+T_c+T_x$	*Nil*

	IdP/ Trustee	$T_{bp}+T_h+T_m+T_i+T_c+T_x$	$2T_h+T_x$	
Authentication	User	$3T_{bp}+2T_h+3T_m+T_i+3T_c+T_x$	$T_{bp}+2T_h+ +7T_c+T_x$	T_m
	Trustee	*Nil*	$T_{bp}+ T_m +5T_c$	
	IdP	$4T_{bp}+4T_h+T_m+2T_c+T_x$	$T_{bp}+ T_m +7T_c$	
Time Complexity		$\mathbf{9T_{bp}+8T_h+6T_m+3T_i+7T_c+4T_x}$	$\mathbf{3T_{bp}+2T_h+3T_m+19T_c+2T_x}$	

Table 4 Running Time Comparisons (In Milliseconds)

Elliptic Curve over Prime Fields in bits	Registration Phase (/300 Records)	Authentication Phase (/41442 Records)
ECDiffieHellmanP521	03.8845493	39.6372597
ECDiffieHellmanP384	04.5017486	40.3048253
ECDiffieHellmanP256	04.9603378	40.7483696
ECDSAP521	05.9775697	41.9618704
ECDSAP384	06.0840950	42.2870329
ECDSAP256	06.7116423	42.9964785

Table 5. Computation Cost Comparisons (In Milliseconds)

Elliptic Curve over Prime Fields in bits	Proposed Scheme	Jia-Lun T et al.[27]	Debiao H et al. [28]
ECDiffieHellmanP521	54.2097481	89.6755458	78.5198765
ECDiffieHellmanP384	54.2339890	90.5648944	79.6564897
ECDiffieHellmanP256	56.9493687	91.5497450	80.6597455
ECDSAP521	57.4219455	92.6512187	81.5649815
ECDSAP384	59.1665572	92.9879895	82.1236587
ECDSAP256	59.9546878	93.2318745	82.8796589

Table 6. Communication Cost Comparisons (In Bits)

Elliptic Curve over Prime Fields in bits	Proposed Scheme	Jia-Lun T et al.[27]	Debiao H et al. [28]
ECDiffieHellmanP521	3296 bits	4320 bits	3296 bits
ECDiffieHellmanP384	3022 bits	4320 bits	3296 bits
ECDiffieHellmanP256	2784 bits	4320 bits	3296 bits
ECDSAP521	3296 bits	4320 bits	3296 bits
ECDSAP384	3022 bits	4320 bits	3296 bits
ECDSAP256	2784 bits	4320 bits	3296 bits

The registration and authentication phase running time of our scheme is recorded in Table IV for different elliptic curves over prime fields. Here, we have considered 300 and 41442 records in registration and authentication phases respectively. The proposed scheme consumes less running time for Elliptic Curves Diffie-Hellman P521. The overall computation cost comparison of our scheme with Jia-Lun T et al.[27] and Debiao H et al. [28] is recorded in Table V for different elliptic curves over prime fields. The proposed scheme consumes less computation cost compare with Jia-Lun T et al.[27] and Debiao H et al. [28] schemes. The overall communication cost comparison of our scheme with Jia-Lun T et al.[27] and Debiao H et al. [28] is listed in Table VI for different elliptic curves over prime fields. The proposed scheme consumes less communication cost compare with Jia-Lun T et al.[27] and Debiao H et al. [28] schemes. Therefore, we can conclude that our proposed authentication scheme is computationally efficient and robust towards various reply and impersonation attacks than the existing schemes.

VIII. CONCLUSION AND FUTURE DIRECTION

In this article, we developed a robust and an efficient mutual authentication model for verifying genuine of communication entities in the cloud using *n*-party Diffie-Hellman bilinear pairing key distribution and random nonce. User credentials and access keys are never revealed to the malicious users. Stakeholders can gain the control over the cloud environment. Experimental results and performance analysis shows that the proposed work is computationally efficient for mutual authentication and robust against the impersonation and ephemeral secret leakage attacks. However, this investigation can be further extended to reduce trustee participation overhead using tokenization techniques.

Declarations

Competing Interests

The authors Mr. Sabout Nagaraju and Dr.S.K.V. Jayakumar declare that they have no competing interests.

Authors' Contributions

Mr. Sabout Nagaraju has made substantial contributions to conception, design, implementation, acquisition of test data, and performed experimental evaluation. Dr.S.K.V. Jayakumar has involved in revising it critically for important intellectual content, supervision of the research work and has given final approval of the version to be published. Both authors read and approved the final manuscript.

Acknowledgements

Thanks to the Pondicherry University administration for providing required hardware and software resources to carry out this work successfully.

REFERENCES

[1] Shane Mitchell, Nicola V et al. "The Internet of Everything for Cities", https://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/everything-for-cities.pdf, June 2013.

[2] Bob Violino, "The dirty dozen: 12 top cloud security threats for 2018", https://www.csoonline.com/article/3043030/security/12-top-cloud-security-threats-for-2018.html, January 5, 2018.

[3] Holger Schulze, "Cloud security report 2018" https://pages.cloudpassage.com/rs/857-FXQ-213/images/2018-Cloud-Security-Report%20%281%29.pdf.

[4] Tim Mather and SubraKumaraswamy, "Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance", http://www.di.fc.ul.pt/~nuno/PAPERS/security3.pdf, 2009.

[5] Jaikumar Vijayan, "Amazon downplays report highlighting vulnerabilities in its cloud service". http://www.computerworld.com/s

/article/9140074/Amazon_downplays_report_highlighting_vulnerabilities_in_its_cloud_service, October 2009.

[6] What is OpenID, OAuth2 and Google Sign In? https://www.youtube.com/watch?v=1M6gqoGiO2s&t=25s, DBA publications, April 20, 2017.

[7] Deep Dive into OAuth for Connected Apps, https://www.youtube.com/watch?v=vlrK3YZ_Fj0.

[8] DBA presenters, "What is OpenID, OAuth2 and Google Sign In?" https://www.youtube.com/watch?v=1M6gqoGiO2s&t=25s, DBA publications, April 20, 2017.

[9] OpenID Foundation, "The OpenID User Interface Extension Best Practices for Identity Providers 2009". [Online]. Available: http://wiki.openid.net/w/page/12995153/Details-of-UX-Best-Practices-for-OPs

[10]J. M. Alves et al. "Multi-Factor Authentication with OpenId in Virtualized Environments", IEEE LATIN AMERICA TRANSACTIONS, VOL. 15, NO. 3, MARCH 2017

[11]Chris Messina, "User Authentication with OAuth 2.0", https://oauth.net/articles/authentication/.

[12]Chris Messina, "OAuth 1.0, OpenID 2.0 and up next: DiSo", https://medium.com/chris-messina/tagged/oauth, 2007.

[13]Aloysius Low, Seth Rosenblatt, "Serious security flaw in OAuth, Open ID discovered", https://www.cnet.com/news/serious-security-flaw-in-oauth-and-openid-discovered/, MAY 2, 2014 4:00 AM PDT.

[14]Hargobind Singh, "Deep Dive into OAuth for Connected Apps", https://www.youtube.com/watch?v=vlrK3YZ_Fj0, October 5, 2015.

[15]Justin Richer, "User Authentication with OAuth 2.0", https://oauth.net/articles/authentication/, 2012.

[16]Pierluigi Paganini, "One oAuth 2.0 hack, 1 Billion Android App Accounts potentially exposed", https://securityaffairs.co/wordpress/53081/hacking/oauth-2-0-attack.html. November 5, 2016.

[17]Merritt Maxim, "Tools And Technology: The Identity And Access Management Playbook", http://www1.janrain.com/rs/253-XLD-026/images/the-forrester-wave-customer-identity-and-access-manageme nt-q2-2017-industry-research.pdf, June 15, 2017.

[18]"Janrain Identity Cloud", http://www1.janrain.com/rs/253-XLD-026/images/janrain-identity-cloud-datasheet.pdf, 2016.

[19]"Identity and Access Management Resource Guide", http://certification.salesforce.com/RG_CertifiedIdentityandAccess ManagementDesigner .pdf, May 25, 2018.

[20]David Goldsmith, "ForgeRock Identity Platform v5.0", https://backstage.forgerock.com/docs/platform/5/Platform-5-Platform-Guide.pdf, 2018. https://www.pingidentity.com/content/dam/ping-6-2-assets/Assets/analyst-reports/en/3208-kuppingercole-solutions-custome r-iam.pdf?id=b6322a80-f285-11e3-ac10-0800200c9a66, March 2017.

[21]John Tolbert, "Ping Identity solutions for Customer Identity and Access Management", https://www.pingidentity.com/content/dam/ping-6-2-assets/Assets/analyst-reports/en/3208-kuppingercole-solutions-custom er-iam.pdf?id=b6322a80-f285-11e3-ac10-0800200c9a66, March 2017.

[22]LoginRadius, "Complete Customer Identity Management" https://www.loginradius.com/press/loginradius-announces-series-a-funding-from-forgepoint-and-microsoft-venture/, July 2018.

[23]Ronny Bjones, "Identity for the 21st Century", https://www.eema.org/wp-content/uploads/bjones.pdf, 2016.

[24]SaboutNagaraju and LathaParthiban, "SecAuthn: Provably Secure Multi-Factor Authentication for the Cloud Computing Systems", Indian journal of Science and Technology, Vol 9(9), March 2016, pp.1-18.

[25]SaboutNagaraju and LathaParthiban, "Trusted framework for online banking in public cloud using multi-factor authentication and privacy protection gateway",Journal of Cloud Computing: Advances, Systems and Applications (2015) 4:22, pp.1-23.

[26]SaboutNagaraju and S.K.V. Jayakumar, "A Novel Approach for Enabling More Accurate Trust and Reputation Mechanisms with an Efficient and High-Security Remote Authentication in the Cloud Computing Environment", Indian journal of Science and Technology, Vol 11(13), April 2018, pp.1-18.

[27]Jia-Lun Tsai and Nai-Wei Lo, "A Privacy-Aware Authentication Scheme for Distributed Mobile Cloud Computing Services", IEEE Systems Journal, Vol. 9, No. 3, Sep. 2015, pp. 805-15.

[28]Debiao He et al., "Efficient Privacy-Aware Authentication Scheme for Mobile Cloud Computing Services", IEEE Systems Journal, Vol. 12, No. 2, June 2018, pp. 1621-31.

[29]James Scott, Drew Spaniel, "In 2017, The Insider Threat Epidemic Begins", https://icitech.org/wp-content/uploads/2017/02/ICIT-Brief-In-2017-The-Insider-Threat-Epidemic-Begins.pdf, February 23, 2017.

[30]L. Gong, R. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols" in Proc. 1990 IEEE Computer Society Symp.Research in Security and Privacy, 1990, pp. 234–246.

[31]M. Burrows, M. Abadi, and R. Needham, "A logic of authentication" ACM Trans. Comput. Syst., vol. 23, no. 5, pp. 1–13, 1989.

[32]S. Pearson, "Taking account of privacy when designing cloud computing services" in Proc. CLOUD ICSEWorkshopSoftw. Eng. Challenges CloudComput., 2009, pp. 44–52.

[33] H. Takabi, J. B. D. Joshi, and G. Ahn, "Security and privacy challengesin cloud computing environments" IEEE Security Privacy, vol. 8, no. 6,pp. 24–31, Nov./Dec. 2010.

[34]Z. Xiao and Y. Xiao, "Security and privacy in cloud computing" IEEECommun. Surveys Tuts., vol. 15, no. 2, pp. 843–859, Jul. 2012.

[35]AmlanJyotiChoudhury, Pardeep Kumar, MangalSain, Hyotaek Lim, Hoon Jae-Lee, "A Strong User Authentication Framework for Cloud Computing", 2011 IEEE Asia -Pacific Services Computing Conference, pp. 110-115, 2011.

[36]J. Yang et al., "A fingerprint recognition scheme based on assemblinginvariant moments for cloud computing communications", IEEE Syst. J.,vol. 5, no. 4, pp. 574–583, Dec. 2011.

[37]SushmitaRuj, M. Stojmenovic, and A. Nayak, "Decentralized access control with anonymous authentication of data stored in clouds", IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 384–394, Feb. 2014.

[38]Neil Zhenqiang Gong and Di Wang, "On the Security of Trustee-Based Social Authentications", IEEE Transactions on Information Forensics And Security, Vol. 9, No. 8, August 2014.

[39]Chin-Ling Chen, Tsai-Tung Yang, Mao-Lun Chiang and Tzay-Farn Shih, "A Privacy Authentication Scheme Based on Cloud for Medical Environment", J Med Syst (2014) 38:143, pp. 1-16, October 2014.

[40]Hong Liu, HuanshengNing, QingxuXiongand Laurence T. Yang, "Shared Authority Based Privacy-Preserving Authentication Protocol in Cloud Computing", IEEETransactions On Parallel and Distributed Systems, Vol. 26, No. 1, pp. 241-251, January 2015.

[41]Jun Zhou, Xiaodong Lin, Xiaolei Dong and Zhenfu Cao,"PSMPA: Patient Self-Controllable and Multi-Level Privacy-Preserving Cooperative Authentication in Distributed m-Healthcare Cloud Computing System", IEEE Transactions On Parallel and Distributed Systems, Vol. 26, No. 6, pp. 1693-1703, June 2015.

Authors' information

Sabout Nagaraju is currently working as assistant professor in Pondicherry University. He is graduated from G.Pulla Reddy Engineering College, Kurnool and did his post graduation at National Institute of Technology, Calicut.
He is pursuing his Ph.D. from Pondicherry University, India. His professional experience spans over 11 years in various Engineering colleges and software industry. His areas of interest include Cloud Computing, Internet of Things and Big Data. He has published eight papers in international journals and seven papers in national/ international conferences.

Dr.S.K.V. Jayakumar is currently working as assistant professor in Pondicherry University. He obtained his B.E in Electrical and Electronics Engineering from Madurai Kamaraj University and did his M.E. in Computer Science and Engineering from Madras University. He has obtained his PhD from Pondicherry University. His teaching experience spans over 19 years and his research interest includes Web Services computing and cloud computing. He has published 29 research papers in International Journals and National and International Conferences.

