International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 18 (2015) pp 39443-39450
© Research India Publications. http://www.ripublication.com

An Elliptic Curve Cryptography based adaptive and secure protocol to access data outsourced to cloud server

D. Sudha Devi
Department of Computing, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, 641 014, INDIA
sudhadevi_cit@yahoo.com

K. Thilagavathy
Department of Physics, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, 641 014, INDIA
thilagavathy@cit.edu.in

Abstract
Cloud computing is renowned for its storage services. Today organizations are interested in outsourcing their data to cloud storages to obtain the significant benefits of cloud services and also to overcome the burden of building and managing the required infrastructure on their own. Being shared and accessed by several users, data stored in third-party cloud storages must be protected against unauthorized access, where, authentication and access control plays a crucial role. The data owner must enhance the security facility for authentication and access control in order to protect data from unauthorized access. In this paper, a combination of secure mutual authentication between data owner, data user and cloud service provider with hierarchical access control protocol based on Elliptic Curve Cryptography is proposed. The proposed protocol receives data request from the users, authenticates the data users in a hierarchy and ensures fine-grained access to the classified encrypted data stored in cloud storage by providing access token.

Keywords-access control, authentication, cloud storage, data outsourcing, data security, elliptic curve cryptography.

Introduction

Cloud computing is fast emerging paradigm which is a ubiquitous internet based computing service. With this technology, users can experience the powerful computing resources and avail ample storage spaces. Today many organizations outsource the data to cloud storages to reap the significant benefits of cloud including reduced cost, scalability, efficiency and service responsiveness to meet the changing business requirements and commercial conditions. Organizations acquire storage and compute resources at minimized operational cost on demand from the service provider without spending much cost in owning the required infrastructure internally. Also a business may opt for outsourcing to experience the innovations of techniques and technologies available with the service provider and also upon outsourcing, an organization can concentrate on prime activities of the business [1,2].

But at the same time, outsourcing sensitive data to third-party cloud storages involves risk in handling the data security issues which includes confidentiality, integrity and availability of data stored in the cloud environment. As the data are stored on third-party premises, data owner lacks full control over their data. Also, since sensitive data are stored on shared servers and are being accessed by various internal and external users, it is mandatory for data owners to build a security system to defend their data against unauthorized access. Authentication and Access control is the fundamental mechanism that provides solution against unauthorized access. To ensure fine-grained access to the outsourced classified encrypted data, an authentication with hierarchical access control protocol using Elliptic Curve Cryptography [3,4,5,6] is proposed in this paper. The proposed protocol is adaptive to the scenario in which, the data owner outsources the classified and encrypted massive data into cloud storage. The data users are also classified into different security classes based on their security clearance which results in a hierarchical tree. The proposed scheme receives data request from users; authenticates whether the user belongs to the corresponding security class; authenticates whether the user has permission to access the requested data and ensures that the stored data is made available only to the authorized users thereby holding fine-grained access control.

Related Work

The purpose of access control technique is to limit the operations that a legitimate user of a system can perform. The importance of authentication and access control and its types are discussed in [7, 8, 9]. Access control techniques comprises of authentication and authorization methods. The different types of authentication methods that could be used for a multilevel security system is enumerated in [10, 11]. Authentication is the process of determining whether someone or something is, in fact, who or what it is declared to be [12]. An access control technique assumes that the authentication of the user has been successfully verified prior to enforcement of access control.

Many researchers have proposed schemes for authentication and access control methods. In this section, schemes related to authentication and key management for hierarchical access control are discussed. Akl and Taylor [13] proposed a cryptographic based solution to solve the problem of access control in a hierarchy. Harn and Lin [14] proposed a cryptographic based solution for the key generation scheme for multilevel data security. Chang and Buehrer [15] proposed solution for hierarchical access control using a one-way trapdoor function. Jeng and Wang [16] proposed a key-management scheme for hierarchical access control using

elliptic curve cryptosystem. Chung et al. [17] proposed an ECC based solution for Access control in user hierarchy. Nikooghadam et al. [18] proposed solution for hierarchical access control using elliptic curve cryptosystem which is assimilated effectively in our scheme. Many authentication schemes have also been proposed by researchers. Peyravian and Zunic [19] proposed methods for protecting password transmission using collision resistant hash function. Liao et al. [20] proposed a password authentication scheme over insecure networks based on the properties of discrete logarithm problem and one-way hash function. Peyravian and Jeffries [21] proposed a secure remote user access over insecure networks using Diffie-Hellman based protocol. Holbl et al. [22] proposed an improved Peyravian-Jeffries's user authentication protocol by employing additional exclusive-or operations. Zhu et al. [23] proposed an improved mutual password authentication scheme over Hwang et al.'s scheme. Lin et al. [24] also proposed an authentication scheme based on one-way hash function over Hwang et al.'s scheme. Observing both Zhu et al.'s and Lin et al.'s scheme, we propose an authentication scheme based on elliptic curve cryptography to overcome few security attacks over their scheme.

Review of Lin et al.'s authentication scheme

Lin et al. proposed an authentication scheme to overcome the security problems that exist in Hwang et al.'s scheme [25].

The Authentication Protocol proposed by Lin et al.[24] is enumerated in this section.

The scheme consists of the following steps:

Step 1: $C \rightarrow S$: id, $\{r_c, pw\}k_s$

A client (C) sends to a server (S), its id, and password pw and a random number r_c by encrypting them using S's public key k_s.

Step 2: $S \rightarrow C : r_s \oplus r_c, H(r_s)$

S with its private key decrypts the received content and retrieves pw and r_c, and compares whether the hashed result of the retrieved pw is the same that is stored in S. Then S performs XOR operation with r_c and a random number r_s and sends the result and the hashed value of r_s to C.

Step 3: $C \rightarrow S$: id, $H(r_c, r_s)$

C performs an XOR operation with $r_c$and $r_s \oplus r_c$ to get r_s. Then the hashed result of the retrieved r_s is compared with the received $H(r_s)$ for equality. C now computes $H(r_c, r_s)$ and sends it to S with its id.

Step 4: $S \rightarrow C$: Access is granted or denied

S computes $H(r_c, r_s)$ with its $r_c$and r_s values, and compares it with the received hash result to determine to grant or deny the C's login request.

Lin et al.'s scheme is vulnerable to the following attacks:

1. Impersonation attack

Consider Step 1 of the scheme, $C \rightarrow S$: id, $\{r_c, pw\}k_s$

Server upon receiving the content from the client, decrypts the content thereby can notice the original pw value. Any privileged or malicious user on the server side can impersonate client knowing the original pw value. The malicious user can replace r_c with his own r_x value and can proceed further. That is, the malicious user can run the authentication protocol with the server in the same manner as that of the original client. Since all the authentication computations will be proceeded without errors, finally, server will grant access permission to client's account which could be used by the malicious user.

2. Insider attack

There are possibilities that a client may use the same identity and pw value for more than one login enrolled with more than one server. Since pw can be retrieved on decryption, any privileged user on the server side knowing the identity and original pw value can try to access other accounts of the client with the same.

Mathematical background of ECC

The public key cryptosystem, Elliptic curve cryptography relies on the believed difficulty of the elliptic curve discrete logarithm for its security. An elliptic curve E over Z_p is a plane curve defined by an equation of the form: $y^2=x^3+ax+b$ where a, b $\in Z_p$ and $4a^2+27b^2 \neq 0 \pmod p$, with a special point O, called the "point at infinity". The set $E_p(a,b)$ consists of all pairs of integers (x,y) that satisfy $y^2 \bmod p=(x^3+ax+b) \bmod p$ together with a point at infinity O. The co-efficient a and b and the variables x and y are all elements of Z_p. Given two points P, Q in $E(Z_p)$, there is a third point, denoted by P+Q on $E(Z_p)$, and the relations commutativity, associativity, existence of an identity element and inverse hold for all P, Q, R in $E(Z_p)$ [3, 4, 5, 6, 26, 27].

The rules for addition over $E_p(a,b)$ correspond to the algebraic technique described for elliptic curves defined over real numbers. For all points P, Q $\in E_p(a,b)$:

1. $P + O = P$.
2. If $P = (x_p, y_p)$, then $P + (x_p, -y_p) = O$. The point $(x_p, -y_p)$ is the negative of P, denoted as -P.
3. If $P = (x_p, y_p)$ and $Q = (x_q, y_q)$ with $P \neq -Q$, then $R = P + Q = (x_r, y_r)$ is determined by the following rules:

$$x_r = (\lambda^2 - x_p - x_q) \bmod p$$

$$y_r = (\lambda(x_p - x_r) - y_p) \bmod p$$

where

$$\lambda = \begin{cases} \dfrac{(y_q - y_p)}{x_q - x_p} \bmod p \text{ if } P \neq Q \\ \dfrac{(3x_p^2 + a)}{2y_p} \bmod p \text{ if } P = Q \end{cases}$$

4. Multiplication is defined as repeated addition; for example $4P = P + P + P + P$.

Proposed System

Preliminaries:

An adaptive and secure authentication and access control protocol using Elliptic Curve Cryptography is proposed in this paper. The proposed protocol is designed using ECC in order to improve and also to overcome the security attacks that exist in Lin et al.'s authentication scheme. The aim in designing the protocol is to ensure authentication and to provide fine-grained access to encrypted data outsourced into untrusted cloud environment. The importance of securing data at rest and a multilevel security framework for the classified data stored in cloud is elucidated exhaustively in [11]. Fig.1 shows the overview of cloud storage scheme.

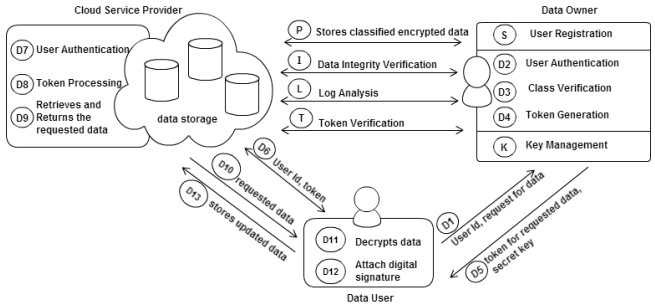

Fig.1. Overview of Cloud Storage Scheme [11]

The Data Owner (DO) outsources the data to cloud storage. The data files are classified and encrypted accordingly before moving into cloud. Initially, Data Users (U) registers with the data owner and thereby they are classified into security classes based on the privileges assigned to them. Whenever a user sends request to access data, the data owner has to authenticate the user and has to verify to which class does the requested data belongs to and to which security class the user exist in.

Once authentication verifications are done, data owner generates token for the requested data and send it to data user. Data user submits the credentials and token to Cloud Service Provider (CSP) for verification. Upon successful verification the user is allowed to access the requested data. Since the cloud storage contains only encrypted data, data owner can ensure confidentiality of the data. Authentication, class verification and token generation based on the type of requested data constricts the access privilege and thereby fine-grained access control is achieved. The goal is to ensure that the stored data is made available only to authorized users with respective access privileges. To accomplish this, an ECC based secure and efficient authentication and access control protocol for user hierarchy is proposed. The following Table.1 depicts the notations used by various entities in the proposed protocol.

TABLE.1. Notations used by various entities in the proposed protocol

Notation	Description
Data Owner (DO)	
d_d	Secret value of DO
P_d	Public key of DO; $P_d=d_dG$
k_i	Secret value of a security class, say C_i
rn_d	Random number chosen from [1,n-1]
T_i	Token for the requested data
M_i	Mapping details for the partitioned data
k_{dc}	Shared key between DO and CSP
E_k	AES symmetric encryption technique
h(x)	Secure one-way hash function
Data User (U)	
Uid_a	Identity of Data User, say user A
d_a	Secret value of User A
P_a	Public Key of User A; $P_a=d_aG$
k_{ad}	Shared Key between Data User A and DO
rn_a	Random number chosen from [1,n-1]
RD	Requested Data
R_i	Request index
Security Class (C)	
d_i	Secret Value for each security class
P_i	Public key of each security class; $P_i=d_iG$

Setup Phase :

Step 1: The DO determines $E_p(a,b)$ an elliptic curve with parameters a, b and q, where q is a prime or an integer of the form 2^m. Also a base point G=(x,y) in $E_p(a,b)$ which is the largest order n such that nG=O is chosen. DO publishes $E_p(a,b)$, G and n to users.

Step 2: The DO classifies the data files, say, $D = \{D_1, D_2, \ldots, D_n\}$ based on their sensitivity and are encrypted and stored in cloud storage.

Step 3: The data users, say, $U = \{U_1, U_2, \ldots, U_n\}$ registers with the DO. DO classifies the registered data users into security classes, say, $C = \{C_1, C_2, \ldots, C_n\}$ based on their security clearance which results in a hierarchical tree structure. The set of security classes can be partially ordered by a binary relation "≤". In this, $C_j \leq C_i$ where i, j ϵ n, C_i is considered to be a security class with higher clearance than C_j, therefore $C_i$are allowed to access the data of C_j and the reverse is not possible.

Step 4: Based on the classification of data files and data users, DO defines access matrices as shown in Table.2 and Table.3 which enumerates the association between security classes and data files, and data users and security classes as follows:

TABLE.2. Security classes-Data files

	D_1	D_2	D_3	D_4	D_n
C_1	1	0	0	1	0
C_2	0	0	0	0	1
C_3	0	1	0	0	0
C_n	0	0	1	0	0

TABLE.3. Data users-Security classes

	C_1	C_2	C_3	C_n
U_1	1	0	0	0
U_2	0	0	1	0
U_3	1	0	0	0
U_4	0	0	0	1
U_n	0	1	0	0

The ECC based Authentication protocol consists of the following phases.

Registration Phase :
$A \rightarrow DO : Uid_a, P_a$

Let user A registers with DO by providing his user identity Uid_a and public key P_a and receives the class details.

Authentication Phase :
Step 1:$A \rightarrow DO : Uid_a \,||\, E_{k_{ad}}(Uid_a, P_i, UV, CV, RD, R_i)$
where $UV= rn_a(d_aP_d)$; $CV= rn_a(d_iP_a)$; $k_{ad}=d_aP_d$

- User A submits his identity Uid_a and the details for authentication and the data to be accessed all encrypted with the key k_{ad}.
- UV is used for user verification which is computed as $rn_ad_aP_d$, where rn_a is a random number, $d_a$is the secret value of user A and P_d is the public key of DO.
- CV is used for class verification which is computed as $rn_ad_iP_a$ where rn_a is a random number, d_i is the secret value of the security class to which the user belongs to and P_a is the public key of user A.
- User A computes encryption key which is the hashed value of the coordinates. $k_{ad}= d_aP_d= d_ad_dG = (x,y) = h(x+y)$.

Step 2:$DO \rightarrow A : (UV + UCV) \,||\, h(UCV)$
- DO receives $Uid_a \,||\, E_{k_{ad}}(Uid_a, P_i, UV, CV, RD, R_i)$;
- Computes $k_{ad}= d_dP_a= d_dd_aG = (x,y) = h(x+y)$; DO uses this key k_{ad} to decrypt the received content.
- Verifies the received user identity with the decrypted user identity.
- DO computes and verifies $\hat{e}(UV, P_i)$ with $\hat{e}(CV, P_d)$based on bilinear pairing proof, to authenticate that the user is an authorized user and to verify his class details and thereby verifies user A has access privilege to the requested data RD.
- Computes $UCV = rn_dd_dP_d$ and sends the user-class verification challenge, $(UV+UCV) \,||\, h(UCV)$ to user A.

Step 3:$A \rightarrow DO : Uid_a \,||\, h(UV,V)$
- User A calculates the final verification V = (UV+UCV)-UV.
- Computes hash code as h(V).
- Compares the computed h(V) with received h(UCV).
- Computes h(UV,V) and sends to DO with Uid_a

Step 4:$DO \rightarrow A$: Access Granted / Denied
- DO computes h(UV,UCV).
- Compares it with received h(UV,V)
- If the comparison satisfies the condition, user A is granted access permission else access is denied.

Token distribution Phase :
Step 1: Token $T_i=Uid_a \,||\, D_i$
Once Access is granted for user A, DO generates a token for the requested data to be submitted to CSP by user A. This token (T) contains information regarding user identity, data files to be fetched, all encrypted with the shared key used between DO and CSP.

Step 2: User A and DO computes the session key as follows.
DO computes the session key as $k_{sk}=rn_dd_dUV$.
A computes the session key as $k_{sk}=rn_ad_aV$.
DO uses this session key for sharing the token for the requested data with A.

Step 3: $DO \rightarrow A : E_{k_{sk}}(R_i, M_i, E_{k_{dc}}(T_i))$
User A can decrypt the received content with the key k_{sk} and checks that he has received the content for the intended request by verifying R_i. DO sends M_i, only if the actual data files are partitioned and stored in cloud storage. With $M_i$user A can merge the partitions to get back the actual data.

Step 4: $A \rightarrow CSP : Uid_a \,||\, E_{k_{dc}}(T_i)$
The encrypted token is submitted to CSP along with the user credentials. CSP verifies the received user identity with his user credential list as well as with the decrypted user identity from the token. If conditions are satisfied, CSP retrieves the data files listed in the token and returns the same to user A.

The following Fig.2 depicts how a DO authenticates data users and the way a data user access the encrypted data stored in cloud storage. Also it shows the method of deriving secret keys for decrypting the requested data.

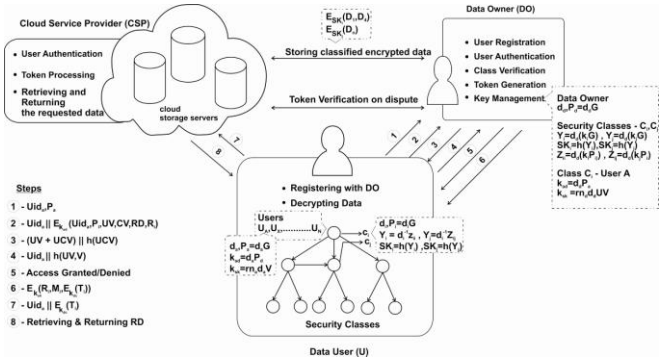

Fig.2. Overview of proposed authentication with hierarchical access control protocol

Key Generation Phase :

Step 1: Each security class, say, C_i selects a secret value d_i, a random number from the interval [1, n−1] and publishes its public value $P_i=d_iG$.

Step 2: For each security class C_i, the DO selects a secret value k_i, a random number from the interval [1,n−1] and computes $Y_i=d_d(k_iG)$. The secret key for each security class is computed as $SK_i= h(Y_i)$, where $h(x)$ is a one-way hash function [3] that is used to transform a point on elliptic curve E_p into hash value. DO uses symmetric crypto algorithms to encrypt the classified data files with the corresponding secret key SK_i for each class before storing into cloud storage.

Step 3: DO computes $Z_{ij}=d_d(k_jP_i)$ and declares it publicly for all security classes C_i that satisfies $C_j \le C_i$ where $1 \le j \le n$.

Key derivation Phase :

Consider $C_j \le C_i$, where $1 \le j \le n$, the security class C_i determines secret key SK_i of its own and for its successor SK_j as follows:

Step 1: Security class C_i determines its secret key as follows:

$$Y_i = d_i^{-1} Z_{ii} = d_i^{-1}(d_d(k_i(d_iG))) = d_d(k_iG)$$

Security class C_i determines its successor C_j's secret key as follows:

$$Y_j = d_i^{-1} Z_{ij} = d_i^{-1}((d_dk_j(d_iG))) = d_d(k_jG)$$

where d_i^{-1} is the modular inverse of d_i in elliptic curve.

Step 2: C_i then computes the key, $SK_i= h(Y_i)$ for its own and $SK_j= h(Y_j)$ for its successor C_j.

Security and Performance Analysis

Security Analysis

The proposed scheme withstands the following attacks which are not possible in the existing schemes discussed earlier.

Masquerade Attack

A masquerade attack is a form of active attack and it takes place when one entity pretends to be an another entity [27].

Attack 1 : Parameters published in the public domain

Data owner publishes Z_{ij} which is declared as a public parameter that can be accessed by the security classes having corresponding security clearance. Since there are opportunities that an attacker can masquerade to be the data owner and can publish some parameters, say Z_{xy} in public domain which is derived using a key, say $Y_y=d_xk_yG$. Data users suppose uses this to derive the secret key of a security class and encrypts their data, then the attacker can be able to decrypt and access the data. Hence the data owner must authenticate the public parameters listed in the public domain. For this an authentication scheme is proposed as follows:

DO publishes the public parameter as $Z_{ij} || E_{k_{ci}}(h(Z_{ij}))$, where $E_{k_{ci}}$ is the shared secret key between DO and security class C_i.

For example, if there exists $C_j \le C_k \le C_i$ where i, k, j ϵ n, C_i is considered to be a security class with higher security clearance than C_k which in turn has higher security clearance than C_j. Then C_i is allowed to access the data of C_k and C_j and its own. Security class C_k is allowed to access the data of C_j and its own. Security class C_j can access the data of its own and not its predecessor security classes.

Now, data owner has to publish the public parameters Z_{ii}, Z_{ik}, Z_{ij}, Z_{kk}, Z_{kj}, Z_{jj}, where security class C_i utilizes Z_{ii}, Z_{ik}, Z_{ij} and derives the secret key of its own and its successors C_k and C_j, security class C_k utilizes Z_{kk}, Z_{kj} and derives the secret key of its own and its successor C_j, security class C_j utilizes Z_{jj} and derives the secret key of its own. So DO publishes the above list of public parameters as follows:

$(Z_{ii} || E_{k_{ci}}(h(Z_{ii}))$

$(Z_{ij} || E_{k_{ci}}(h(Z_{ij}))$

$(Z_{ik} || E_{k_{ci}}(h(Z_{ik}))$

$(Z_{kk} || E_{k_{ck}}(h(Z_{kk}))$

$(Z_{kj} || E_{k_{ck}}(h(Z_{kj}))$

$(Z_{jj} || E_{k_{cj}}(h(Z_{jj}))$

Before utilizing the public parameters, security classes must verify whether the public parameters are published by DO. Since the hash code of each public parameter is encrypted with the shared key between DO and corresponding security classes, the security classes thereby ensures that the parameters are from authenticated DO. Each security class can decrypt the content with the shared key of its own. Then the decrypted hash value is compared with the hashed result of the published parameter. If conditions satisfy, security classes are ensured with authentication and can utilize the public parameter for further processing. The adversary cannot publish his parameter as the shared key between data owner and each security class is unknown.

Attack 2 : Token processing phase of CSP

In the token distribution phase, DO issues token for the data requested by the user. In the content of token along with the data files to be fetched, identity of the user who requested the data is also included to avoid masquerade attack. For example, any authorized user, say B in the security class if accidently

knows the token of its predecessor may try to submit the token along with his credential to the CSP. In the absence of user identity in the token content, CSP would check the credential with its user database and since B is also according to CSP is an authorized user, will retrieve and returns the requested data to user B. Since user identity is included in the token, now CSP checks the submitted credentials not only with his user database but also verifies with the received identity which helps to avoid masquerade attack. That is, in the token, since user A's identity is present, it mismatches while checking and CSP can intimate this token dispute to DO to proceed further.

Attack 3: Masquerade to know secret value

An intruder pretends like a server to know the user's secret value. But the user's secret value is not stored explicitly, instead the user is verified using the public key d_aG. To retrieve d_a from d_aG is not possible for any intruder. Also the symmetric key $k_{ad}=d_aP_d=d_dP_a$ cannot be computed without knowing DO's secret value d_d or user's secret value d_a. Therefore the intruder cannot proceed with step 2 and hence the masquerade attack results in failure.

Replay Attack

Replay attack is the passive capture of information by an adversary and subsequent transmission of it to produce an unauthorized effect [27].

In the proposed scheme, consider an adversary captures the earlier sent message $Uid_a \| E_{k_{ad}}(Uid_a, P_i, UV, CV, RD, R_i)$ and submits to data owner. Since throughout the authentication scheme, all the steps involves with the secret value of either the client or the data owner, an adversary should know the values of d_a and d_d which is impossible. If d_a is known then only the adversary can compute UV from (UV+UCV) and in turn to verify h(UCV). To get d_a from d_aG, the adversary should solve discrete logarithm problem for elliptic curves. Prior to this, since the authentication message is encrypted with k_{ad}, the adversary has to find and compute $k_{ad}=d_aP_d=d_dP_a= d_ad_dG$ which is highly impossible and hence the adversary cannot proceed further and the replay attack results in failure.

Impersonation Attack

An impersonation attack is an attack in which an adversary successfully assumes the identity of one of the legitimate parties in a system or in a communications protocol [28].

A "replay attack" can be considered as a specific kind of impersonation. Consider an adversary tries to impersonate a legitimate client. The adversary performs a replay attack and sends the authentication message to server. It is not possible for the adversary to determine UV or UCV values without the knowledge of k_{ad} for decryption and d_a and d_d for computing UV or UCV. Any submission of wrong computation will be identified on the data owner computation and accordingly access permission will be denied to the adversary.

Insider Attack

An insider attack is made by an entity with authorized system access which is perpetrated on a network or on a computer system [29, 30].

Insider attack on the data owner side:

The privileged user on the data owner side cannot determine the secret value d_a of the client from d_aG, since it requires to solve elliptic curve discrete logarithm problem. Without knowing d_a, the privileged user cannot compute the shared key k_{ad}, and hence cannot impersonate the legitimate client and the insider attack results in failure.

Insider attack on the CSP side :

The data owner stores the classified data in encrypted form. The key management is done by the data owner and the data user can compute keys for decryption. Any privileged user on the CSP side, without the knowledge of the type of data stored and the method for deriving keys is impossible to derive any information regarding the data stored in cloud servers.

Table.4 gives the summary of security analysis between existing schemes and the proposed protocol.

TABLE.4. Security Analysis

Security Attacks	Lin et al.'s scheme	Zhu et al.'s scheme	Proposed protocol
Masquerade Attack	Yes	No	No
Impersonation Attack	Yes	Yes	No
Insider Attack	Yes	No	No
Replay Attack	Yes	No	No
Cryptosystem used	Hash based	Hash based	ECC based
Session Key used	No	No	Yes

Performance Analysis

The Elliptic curve cryptography has received significant attention by academicians and researchers due to its high security level with small key size, high performance and low computational cost. The proposed scheme is based on elliptic curve cryptography. The computational overhead involved in this scheme is discussed in this section. The computational cost includes both the authentication scheme and the hierarchical access management based on Elliptic Curve Cryptography [5] is described below. Table.5 depicts the notations used in the performance analysis.

TABLE.5. Notations used in performance analysis

T_{EC_MUL}	Time to execute multiplication in an elliptic curve point
T_{EC_ADD}	Time to execute addition of two points in an elliptic curve
T_{EC_SUB}	Time to execute subtraction of two points in an elliptic curve
T_{HASH}	Time to execute hash function
T_{INV}	Time to execute modular inverse

Total computational amount for authentication scheme is given as follows:

$T_{AUTH} = 8.T_{EC\_MUL} + 6.T_{HASH} + T_{EC\_ADD} + T_{EC\_SUB}$

where,

- $8.T_{EC\_MUL}$ includes the amount for computing verification parameters UV which is $(2.T_{EC\_MUL})$, CV which is $(2.T_{EC\_MUL})$, computing encryption key k_{ad} which is $(2.T_{EC\_MUL})$, computing verification parameters, UCV which is $(2.T_{EC\_MUL})$.
- $6.T_{HASH}$ includes the amount for computing 2.h(x+y), h(UCV), h(V), h(UV,V) and h(UV,UCV).
- T_{EC_ADD} is the amount for computing UV+UCV.
- T_{EC_SUB} is the amount for computing V.

Total computational amount for hierarchical access scheme is given as follows:

$T_{HAC} = 2.n.T_{EC\_MUL} + 2.n.T_{HASH} + n.T_{INV} + 2.\sum_{I=1}^{n} (v_i+1) .T_{EC\_MUL}$

where

- $2.n.T_{EC\_MUL}$ includes the amount for computing public key P_i for n security classes which is $n.T_{EC\_MUL}$ and computing Y_i for n security classes is $n.T_{EC\_MUL}$.
- $2.n.T_{HASH}$ includes the amount for computing secret keys SK_i by the data owner for n security classes which is $n.T_{HASH}$ and computing secret keys SK_i for n security classes by the user which is $n.T_{HASH}$.
- $n.T_{INV}$ includes the amount for computing inverse d_i^{-1} for n security class keys which is $n.T_{INV}$.
- 2. $\sum_{I=1}^{n} (v_i+1) .T_{EC\_MUL}$ includes the amount for computing Z_{ij} for key generation which is $\sum_{I=1}^{n} (v_i+1) .T_{EC\_MUL}$ and computing to derive all values of the secret keys which is $\sum_{I=1}^{n} (v_i+1) .T_{EC\_MUL}$. Here, v_i refers to the number of predecessors of each security class.

Therefore,

$T_{HAC}=2.n.T_{HASH} +(n + \sum_{I=1}^{n} (v_i+1)).2. T_{EC\_MUL}+n.T_{INV}$

Total computational amount for the proposed scheme = $T_{AUTH}+T_{HAC}$

Conclusion

A combination of secure authentication with efficient hierarchical access protocol based on Elliptic Curve Cryptography is proposed to access the encrypted data stored in cloud storage. In the proposed scheme, the cloud storage is used to store massive volume of data that is classified and encrypted based on its sensitivity. The data owner receives data request from users, authenticates and verifies whether the user has access privilege and issues token for the requested data if the user is an authorized user. In this proposed scheme, an adversary cannot impersonate the legitimate user, because the adversary cannot obtain any information during the authentication phase since the protocol is based on ECC and all communications between data owner and user are protected with encryption technique. Also to access the data, the authorized users derive secret keys using their private key of the security class. Here too an adversary cannot obtain any information regarding the secret keys of the security class. Since the data are stored in encrypted form, any privileged user on the cloud service provider side also could not obtain any information regarding the stored data. At any point of time, an adversary trying for any kind of attack has to solve the elliptic curve discrete logarithm problem in order to obtain secret values used in this proposed scheme. The strength of the proposed scheme is that it is designed based on the strength of ECC. Thus the proposed ECC based authentication with hierarchical access control protocol is a secure and efficient protocol which provides fine-grained access to the data stored in cloud environment.

References

[1] Singh, S., and Zack, M. H., 2006, "Information Technology Outsourcing: Reducing Costs or knowledge?," OLKC 2006 Conference, University of Warwick, Coventry, 20th-22nd March 2006.

[2] Tayauova, G., 2012, "Advantages and disadvantages of outsourcing: analysis of outsourcing practices of Kazakhstan banks", Procedia-Social and Behavioral Sciences, (41), pp. 188-195.

[3] Menezes, A.J., Van Oorschot, P.C., and Vanstone, S.A., 1997, "Handbook of Applied Cryptography", CRC Press Series on Discrete Mathematics and its Applications, CRC Press, Boca Raton, FL, pp. 321-376.

[4] Hankerson, D., Menezes, A.J., and Vanstone, S., 2004, "Guide to Elliptic Curve Cryptography", Springer-Verlag, New York, USA, pp.78-80.

[5] Koblitz, N., Menezes, A., and Vanstone, S.A., 2000, "The state of elliptic curve cryptography," Designs, Codes and Cryptography (19), pp. 173-193.

[6] Miller, V.S., 1986, "Use of elliptic curves in cryptography," Advances in Cryptology CRYPTO 85, Lecture Notes in Computer Science 218, pp. 417-426.

[7] Sandhu, R. S., and Samarati, P., 2000, "Access control: principle and practice," IEEE Communications Magazine, 32(9), pp. 40-48.

[8] Jung, Y., and Chung, M., 2010, "Adaptive security management model in the cloud computing environment," Proceedings of the 12th International Conference on Advanced Communication Technology, Busan, The Republic of Korea, February, pp. 1664-1669.

[9] Purohit, V., 2008 "Authentication and access control—the cornerstone of information security," TrianzWhite Paper.

[10] Sudhadevi, D., Thilagavathy, K., Vaghula Krishnan, S., Harish, S., and Srinivasan, R., 2014, "Integrating OTP authentication service in OpenStack," Advanced Materials Research, 984(985), pp. 1309-1317.

[11] SudhaDevi, D., and Thilagavathy, K., 2015, "An Adaptive Multilevel Security Framework for the Data Stored in Cloud Environment," The Scientific World Journal, (2015), pp. 1-11.

[12] Patel, D. R., 2008, "Information Security—Theory and Practice", Prentice Hall, New Delhi, India, pp. 107-169.

[13] Akl, S.G., and Taylor, P.D., 1983, "Cryptographic solution to a problem of access control in a hierarchy" ACM Transaction on Computer Systems, 1(3), pp.239-248.

[14] Harn, L., and Lin, H.Y., 1990, "A cryptographic key generation scheme for multilevel data security" Computer Security, (9), pp. 539-546.

[15] Chang, C.C., and Buehrer, D.J., 1993, "Access control in a hierarchy using a one-way trapdoor function" Computers and Mathematics with Applications, 26(5), pp. 71-76.

[16] Jeng, F.G., and Wang, C.M., 2006, "An efficient key-management scheme for hierarchical access control based on elliptic curve cryptosystem," The Journal of Systems and Software, (79), pp. 1161-1167.

[17] Chung, Y.F., Lee, H.H., Lai, F., and Chen, T.S., 2008, "Access control in user hierarchy based on elliptic curve cryptosystem" Information Sciences, (178), pp. 230-243.

[18] Nikooghadam, M., Zakerolhosseini, A., and Moghaddam, M.E., 2010, "Efficient utilization of elliptic curve cryptosystem for hierarchical access control," The Journal of Systems and Software, 83(10), pp. 1917-1929.

[19] Peyravian, M., and Zunic N., 2000, "Methods for protecting password transmission," Computers and Security, 19(5), pp. 466-469.

[20] Liao, I., Chi Lee, C., and Hwang, M., 2006, "A password authentication scheme over insecure networks," Journal of Computer and System Sciences, (72), pp. 727-740.

[21] Peyravian, M., and Jeffries, C., 2006, "Secure remote user access over insecure networks," Computer Communications, 29(5), pp. 660-667.

[22] Holbl, M., Welzer, T., and Brumen, B., 2008, "Improvement of the Peyravian-Jeffries's user authentication protocol and password change protocol," Computer Communications, (31), pp. 1945-1951.

[23] Zhu, L. Yu, S., and Zhang, X., 2008, "Improvement upon Mutual Password Authentication Scheme", International seminar on Business and Information Management, vol. 1, pp. 400-403.

[24] Lin, C.L., and Hwang, T., 2003, "A password authentication scheme with secure password updating," Computers and Security, 22(1), pp. 68-72.

[25] Hwang, J.J., and Yeh, T.C., 2002, "Improvement on Peyravian-Zunic's password authentication schemes," IEICE Transactions on Communications, E85-B(4), pp. 823-825.

[26] King B. 2009, "Mapping an Arbitrary Message to an Elliptic Curve when Defined over GF(2n)" International Journal of Network Security, 8(2), pp. 169-176.

[27] Stallings, W., 2006, Cryptography and Network Security—Principles and Practices, Prentice Hall, New Delhi, India, pp. 13-350.

[28] Adams, C., 2011,"Impersonation Attack" Encyclopedia of Cryptography and Security, Springer US, pp. 596.

[29] Claycomb, W. R., and Nicoll, A., 2012, "Insider threats to cloud computing: directions for new research challenges," Proceedings of the 36th IEEE Annual International Computer Software and Applications Conference (COMPSAC '12), July, pp. 387-394.

[30] Yusop, Z.M., and Abawajy, J.H., 2014, "Analysis of Insiders Attack Mitigation Strategies," Procedia-Social and Behavioral Sciences, (129), pp. 581-591.