Journal of Engineering and Applied Sciences 12 (24): 7619-7632, 2017
ISSN: 1816-949X

A Hybrid RBAC and Chaos Access (HRCA) Control Mechanism for Cloud Computing Environment

[1]G. Venifa Mini and [2]K.S. Angel Viji
[1]Department of Computer Science and Engineering, Noorul Islam University, Kumaracoil, India
[2]Department of Computer Science and Engineering, College of Engineering, Kidangoor, India

Abstract: Access control techniques are the very important aspect of cloud architecture, through which data publishing can be regulated to the legitimate users. But at the same time it should not consume huge computational load which may burden the user. In the conventional approach, the data is encrypted as per the access structure and the user able to decrypt only the attribute set matches. However, they require huge computational requirement in terms of space and processing capability. Problem such as preventing cloud server from learning user access privilege related information is also important. Providing support to a continuously varying user dynamics with dynamic user grant and revocation is essential. In Hybrid RBAC and Chaos Access (HRCA) control mechanism, the control is employed over an already encrypted data. Also, the user is allowed to access the data based on the access structure assigned to his role at the server itself. It will warrant security from non-trustable servers. This scheme is implemented on a three layered approach by separating key manager from cloud server, in order to remove key escrow and arbitrary-state attribute expression parities in existing methods. To ensure greater flexibility in case of emergency, a break-the-glass data access approach also included for data availability without any denial of service. Simulation result shows that the approach derive minimal disk space, processing time, access delay time, energy expenditure and cost than RBAC, MAC, DAC.

Key words: Access control, cloud, chaos, privacy, expression, flexibility

INTRODUCTION

Cloud computing is rapidly replacing conventional means computing nowadays (Armbrust *et al*., 2010). Cloud computing is allowing cloud users to avail computing hardware, software and infrastructure platforms at predominately lower costs (Marston *et al*., 2011). Data storage and publishing it to select users is the most availed cloud service (Hawang and Li, 2010). Cloud technology enables the service provider to install his computing capacity in more convenient geographical location with reference to lower costs, tax burden and minimal compliances (Ali *et al*., 2012). On other hand, a cloud user residing at location where data hosting services are costlier can avail the service from the cloud server hosted by cloud service provider (Zhou *et al*., 2010). Normally, a cloud service provider shares his server with a number of users on rental basis (Marston *et al*., 2011). The service provider also leases his storage and processing capability to his users (Hariri *et al*., 1999). A cloud user normally stores data in cloud servers for the purpose of fulfilling his client's data needs or to allow users to access his data (Dawn *et al*., 2012).

The number of users needed to access the data belongs to Data Owner (DO) changes dynamically over time (DiVimercati *et al*., 2007). The DO also concerns about sharing particular set of data to selected group of users. On other hand he may need to hide some kind of data from one group users for which he allows access to other group (DiVimercati *et al*., 2007). Due to the dynamic nature of user patterns, the DO needs to dynamically update the roles and permissions of users to access data. Access control techniques are the very fundamental aspect of cloud architecture through which data publishing can be regulated to the permissible users (Zomaya and Lee, 2012). But at the same time it should not consume huge computational load which may burden the user.

The existing access control mechanisms strive to achieve greater flexibility, scalability, dynamic support, security and fine-grained access. Encryption based access control techniques able to achieve data security along with privacy protection (Damiani *et al*., 2007; Osborn *et al*., 2000; Rost *et al*., 2014). However, they require huge computational requirement in terms of space and processing capability (Foster *et al*., 2008). Problem

Corresponding Author: G. Venifa Mini, Department of Computer Science and Engineering, Noorul Islam University, Kumaracoil, India

such as preventing cloud server from learning user access privilege related information is also important. Providing support to a continuously varying user dynamics with dynamic user grant and revocation is essential (Foster *et al*., 2008).

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is one of the main access control scheme that was used to provide an attribute based access control (Bethencourt *et al*., 2007). The subject's cipher key is defined with an attribute set and the encrypted object is linked with an access structure (Victor *et al*., 2010). The owner of data is able to define the roles and permission in the access structure at the time of storage (Zhou *et al*., 2011). A subject may access and read the content of data only when his attributes match the one defined by the data owner. The cost of storing and processing is high for CP-ABE (Sekhar *et al*., 2012).

In the conventional approach the data is encrypted as per the access structure and the user able to decrypt only the attribute set matches. In our approach access control is employed over an already encrypted data. Also the user is allowed to access the data based on the access structure assigned to his role at the server itself. By limiting the user to access data based on his role, the problem of exposing the accessing structure is limited. Since, the access control is employed on an already encrypted data it will warrant security from non-trustable servers.

Suppose n represents a normal access structure then nr is the proposed scheme in which access is given based on structure on an already encrypted data $ rather than encrypting the raw data " and allowing users to decrypt it based on his attribute set. In the current scenario access control encryption can be described as n(") which is changed to nr($). The access structure nr is not known to the key manager hence any malicious intention of the key manger is void. At the same time, the keys used to encrypt $ are not known to the server.

In this study, we propose a novel access control technique by combining RBAC and chaos. The proposed scheme will also be analyzed on cost and computational requirement front. It is a cost effective access control technique with very less computational requirement that capable of achieve user dynamic portability, fine grainedness and flexibility is essential in the current scenario. Further, security proofs are also been validated to understand the capability of this technique to ensure data security.

Proposed scheme: We propose a novel access structure based hybrid RBAC and chaos access control encryption schema. This scheme is implemented on a three layered approach basis in order to remove the parities in existing methods. At the first layer the key manager is a separate entity that will manage keys, user control and administration such as user access grant and revocation, etc. At the second layer, the cloud server will be notified of the role of user dynamically from time to time. The cloud server will treat the user based on his role and will implement the access structure and allow access to data. To ensure greater flexibility in case of emergency a break the glass data access approach also included. problems: key escrow and arbitrary-sate attribute expression. Although, the key manager is aware of the keys, it is not aware of the access structure defined that is located in the cloud server that avoids any possible attack. We summarize our proposed scheme as follow as.

A separate and dedicated Key Management (KM) module is included that is not allowed to access the access structure and privileges of roles. The access structure nr and privileges of each role is only known to DO and cloud server.

The Cloud Server (CS) is unaware of data content since it is given with only encrypted form of data. Also, the identifiers even during request handling will be in encrypted form preventing the cloud server from knowing the underlying data. The key used to encrypt the data is not known to the cloud server as it is managed separately by the KM Module. This paves way to achieve data security and enhanced privacy control.

We employ an access structure based data control over an already encrypted data rather than encrypting data in accordance with the policies that summarily reduces computational space and complexity. Even the reduced cost of computation is shared among access control and data security module.

We establish the proposed scheme on a test bench basis for evaluation in terms of storage requirement, computational load, cost , finegraineness and security.

Prerequisites

Access structure: Let {D1, D2, ..., Dn} can be assumed as a group of subjects. The group Mf{D1, D2, ..., Dn} is monotone when œF, J: if FOM and F f J, then JOn; Hence, an access structure nr is a group M of non-empty subsets comprising {D1, D2, ..., Dn}. In other words, it can be written as Mf2 {D1, D2, ..., Dn} { i }. The sets included in M are called authentication sets else they are known as non-authentication sets. Here, M will have the authorized set of identifiers.

Bilinear mapping: For a prime order S, we assume G_O and G_T were the multiplication cyclic groups. The originator of G_O is g. Then a bilinear mapping e: $G_O \times G_O$ should follow the below characteristics:

- Bilinearity: œu, v0G_O and a, b0Zp, it can be said as $e(u^a, v^b)^{ab} = e(u, v)^{ab}$
- Non-degeneracy: ›u, v0G_O such that e(u, v)…1
- Computability:œu, v0G_O there exists an algorithm to arrive e(u, v)

Access tree: Let nr be the proposed access structure tree with the root node being represented as K. For better understanding about the nr their components and roles are given as.

The P is attributed to the node of nr. When P is representing the leaf node then P indicates the attribute in encrypted form. When P is representing a non-leaf node, it denotes a functional gate such as "NOT" and "OR". Further: num_P is attributed to P's child in the structure nr.

Also, k_P describes the threshold of P where $0<k_P<num_P$, if $k_P = 1$ and P is a non-leaf, it denotes OR gate. At the same time $k_P = num_P$ and P is non-leaf node then it is an AND gate:

- Parent (P) denotes the parent of P in nr
- Attr (P) indicates the attribute of P in nr
- Index (P) provides for the identifier value of the P in nr. Each P is attributed with an exclusive identifier

The nr_P indicates the sub tree of nr roots out from point P. When a group of attributes in encrypted form represented as É valid to the sub tree nr_P, it is collectively denoted as $nr_P(É) = T$. On other hand $nr_P(É)$ is arrived when P is a non-leaf node and if k_P child then return 'T'. Similarly if P is a leaf point then $nr_P(É)$ provides 'T' only when weight of attr (BP)$weight of attr (P).

A number of roles 'R' are created with each 'R' having their own set of access structure nr and each nr is defined with a permissions set {"1, "2, ...}. The root node K will have an exclusive identifier g_i such that for each role, R f g_i. Each and every user has their own role prior to access the data been validated with g_i and redirected to access their corresponding access structure nr. The nr can be detailed as to nr($) indicating the tree consists of data and attributes in the encrypted form.

The access structure will allow the users to access the data as per the policy structure defined to him. A dedicated access tree with more privacy control nr(U) is set for unauthorized public users. Although, the public users are treated like that of any role, they are restricted with better privacy and security control. Data is masked in its encrypted form prior to send to a user belongs to the group U.

MATERIALS AND METHODS

Proposed architecture of the scheme: The architecture and working of the proposed scheme is detailed in Fig. 1 and 2, respectively. The scheme architecture consists of four modules namely the Data Owner (DO), Key Manager (KM), Cloud Service Provider (CSP) and the Data Users (DU). The assumptions underlying with the each module and their respective working is described in this study.

Data owner: The DO is the ultimate owner of the data located in the cloud server, he hired the cloud server for the purpose of data storage from the CSP. The DO is one who decides data access policies, roles R, privileges and access structure nr. The DO also enables new users to access the data from the cloud server by granting them permission. Further he may revoke a permission already granted to a user. KM is assumed as a semi-trustable segment in the architecture provided that it is honest but curious one. It indicates that the KM will execute the assigned tasks without fail and provide the expected outcomes in terms of performance. But it will be curious about the data stored and may gather as much of confidential and important information. In a cloud architecture it is not possible for the DO to be in online all the time. Hence, DO dedicates some functions related to user management with the KM. One function of the KM is to authenticate the DU based on the details received from DO. Once authenticated proper role will be assigned to the particular DU and his key termed as R_u will be shared with the CSP. The KM will also share the DO's key R with the DU so that he will be able to decrypt the data. The KM updates itself dynamically with the user management related data received from DO.

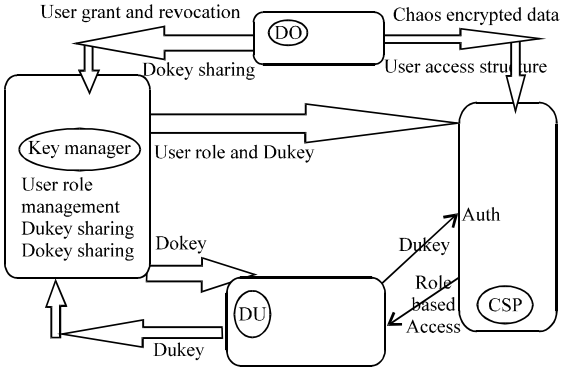

Fig. 1: Architecture of proposed access control scheme

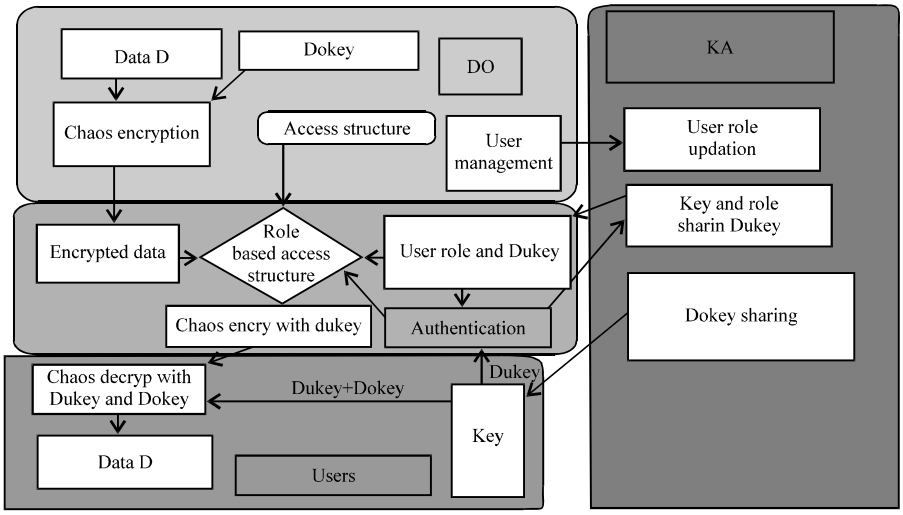

Fig. 2: Working of the proposed scheme

Cloud Service Provider (CSP): It is the principle organ of the architecture where the encrypted $ data belongs to DO is stored. The CSP is assumed as a semi-trustable segment which is honest but curious. The CSP will perform the tasks assigned to it honestly but it may try to access the data contents. A number of services are offered by CSP namely data storage, processing and data transmission. The CSP apart from storing the data belongs to DO, it also implements the access structures defined for each roles. On receiving the authentication role information and DU's key R_u, the CSP will pick-up a access tree nr matching the user role R. Then the CSP encrypts the already encrypted data $ with the R_u. The encrypted data denoted as $R_u(\$)$ must satisfy the access structure set nr. The CSP will also carry out data Anonymization prior to sending the data to the user if he is a public user and have no authorization.

Data Users (DU): The data users are main consumers of data. They include both authorized and unauthorized users with varying roles and permissions. They authenticate with KM with their R_u after that they obtains R KM. The R is one of the key required to decrypt data received from CSP. The CSP is contacted by DU with his R_u. The CSP will authenticate the user with the details it received from KM. Once authenticated the role R as given by the KM will be assigned to DU. After that the DU is given privilege to access the data in the server with the access structure allotted to him. Once the data is requested, CSP will encrypt it with R_u and sends it to the DU. The DU on receiving the data will decrypt the data with R_u and R. If the DU is a non-registered user or public user he will be able to access the data with the access structure nr(U).

Architecture of the proposed technique: The proposed technique has multiple stages which are detailed in the following study.

Stage 1; Key generation, sharing and role assigning): The first stage of the architecture is known as the initialization stage. In which all the modules involved in the setup will initialize themselves.

Data owner (R^k): Let us say a state of incidence k occurred in which the DO generates a key series R with a security parameter value 8 sharable to his users. The key series R will be used to encrypt the data as R($). The encrypted data shall be send to the CSP at the state k. Also the security value 8 will be encrypted with an encryption coding mechanism M that is known to the KM alone. Thus, the security value M(8) is securely transmitted to the KM. The data owner also defines the access structure nr and sends it to the CSP. This is also the state where a DO originally creates the set of users (users set) authorized to access data their Roles R and their corresponding access structure nr.

At the state k, the KM will receive the M(8) and store it in the encrypted form. The CSP will store the data received R($)and will keep it in the encrypted form as it is unaware of the key series.

State 2; User authentication: A second stage state l occurs when either an authorized or unauthorized public user intends to access the data belongs to DO. At this state the user contacts the KM with a security parameter value 8_u which as a identifier used by DO to distinguish between the users. The KM establishes the authenticity of user by matching the 8_u in the users set. Once the user

is authenticated the KM will decrypt 8 by following the reverse of the encoding mechanism M. Then the security value of DO 8 is be shared with the DU by encrypting it with 8_u.

The $8_u(8)$ will be transmitted to the DU. Simultaneously the KM will share the 8_u with the CSP along with the Role R information belongs to the corresponding user.

Stage 3; Data encryption and publishing: This is the final and crucial stage at which the data will be published to the data users with proper encryption and access control mechanisms. It is denoted as m.

The DU will contact the CSP and will establish its authenticity by providing 8_u. The CSP will use the 8_u to match with the information received from KM. This state m is a session based state in which the information shared by KM to the CSP is valid only for a particular time t. Once, the CSP successfully authenticated the user it will match the role information R received from KM to identify the access structure the user belongs to. Once the access structure belongs to role is identified, the CSP will encrypt the already encrypted data R($) with a key series generated with security value 8_u.

The series is generated with chaos based theory and 8_u is given as the initial condition required to generate the key serials. For scalar data the key serials are generated with Logistics mapping technique with $L_{n+1} = Al_n (1-L_n)$, A , [0, 4], L_n , [0, 1] where A is the logistic parameter, which decide the distribution patterns of equation. L_n , [0, 1] is the condition of a chaotic state that is sensitive to the initial condition L_o. When 3.5699456<A<4. Similarly, the key serials required to encrypt multimedia data such as image are generated with Henon mapping. The procedure is as follow as:

$$x_{n+1} = 1\text{-}ax_n^2 + b \quad (1)$$

$$y_{n+1} = x_n \quad (2)$$

Where:
a = 1.4
b = 0.3
n = 1, 2, …

After generating the key serials the encrypted data is 8_u(R($)) with a key series generated with security value 8_u. This data 8_u(R($)) is mapped to the access structure nr in real time basis and the user is allowed to access the data. Every data query is mapped with the user's corresponding access structure nr and encryption is applied prior to sending to the user. If the user belongs to the set nr(U) then the CSP implements data masking on the data along with enforcing the corresponding access structure. Post encryption, the CSP sends data to the DU.

Stage 4; Data decryption: On receiving the data the DU will use his 8_u to generate key serials required to decrypt 8_u(R($)) followed by decryption with the DO's key series generated with R in order to obtain the original data $.

Proposed hybrid chaos-RBAC scheme: The detailed design of the proposed hybrid chaos-RBAC based access control scheme is given in this study. They include initialization, data creation, role definition, access structure definition, key sharing, data user authentication, data access, data transmission, user role revocation, user role change and data masking.

Initialization: For a prime order S, we assume G_O and G_T were the multiplication cyclic groups. The originator of G_O is g, then a bilinear mapping e: $G_O \times G_O$ 6G_T should follow the bilinearity œu, v , G_O and a, b , Z_p it can be said as e(u^a, v^b) = e(u, v)^{ab}.

Along with this the group of subjects marked into a group {D1, D2, ..., Dn} for which œF, J: if F , M and F f J, then J , n. Here, the access structure nr includes the group of subjects {D1, D2, ..., Dn} that are non-empty. The set of subjects and their respective authorization is mathematically M f 2{D1, D2, ..., Dn}{ i }. All the identifiers in the defined set are authenticated in nature and any element out of this set is non-authenticated.

DO initialization: The DO initialization is carried out with a security value 8. DO applies 8 to a chaos theory based key series generator. The DO key thus generated is a highly random polynomial series R with initial value as 8. Where as R = {8}.

KM activation: The KM activates by receiving M(8) and M(R, R_u) and store it in the encrypted form:

$$\text{Dokey} = \{\lambda\},\ \text{User role} = \{R \cup \psi_u\}$$

CSP activation: The CSP activation is started when the KM notifies it regarding the role and user key 8_u. The CSP matches the Role R with the set of allowed privileges set. Once the role is matched, it will create the key series for R_u KM activates by receiving M(8) and M(R, R_u) and store it in the encrypted form:

$$\text{Dukey value} = \{\lambda_u\}$$
$$\text{Dukey} = \{\psi_u\}$$

Data preparation: Prior to transmitting the data to the CSP, the data is encrypted by the DO. Hence, the data stored in CSP is in encrypted form. DO encrypts the data $ from a key series R. On other hand when a request for access is raised by a DU, the CSP will execute the following steps:

- CSP encrypts data as $R_u(R(\$))$. Here, R_u is a key series generated with chaos theory with initial value as 8_u
- Simultaneously, the access structure nr corresponding to the user role R is matched from the collections of nr
- The CSP encryption of data is done on an already encrypted data

In accordance with the user's request for data access, the CSP matches the data identifier received from user with the stored data. It then encrypts it using R_u. The recurring requests are processed only based on the access structure. When a request breaks the access structure it will be readily denied by the CSP.

CSP data send (R_u, $R_u(R(\$))$): Initially a polynomial Q_P is selected for each node P including the leaf nodes for the access structure nr. Starting from the root node K, the polynomial Q_P is obtained in a up to down approach. Also, each and every node in the access structure nr a degree of polynomial d_P is set with K_P-1 where K_P denotes the threshold value. For each non-root node P, it sets as QP = Q parent (P) (index (P) and randomly chooses K_P.

New user authorization: When a user needs to have a full-fledged data access, he need authorization to do so. For this purpose, the user needs to register himself with the cloud system. The enrollment is done by the DO through the KM. An user desirous of joining the cloud system sends his request to the KM. The KM matches the identity of the user with the general set received from the data owner. The general set of users as send by the DO to KM will have a generalized user pattern. If the new user willing to join is one among the predefined user set When a user wants to join the cloud system, KA first accepts if the user forms a member of the set {D1, D2, ..., Dn}. When the register is accepted, this is followed by allocating role to the user by the KM from the role set. When a user does not belong to the authenticated user sets, he refer the user to DO. The DO decided thereupon either to include the new user under authenticated user set or not during his immediate next cloud session. Once, KM allows the new data user to register himself, it is followed by role allocation. After the registration, if the new user desirous of access data from the CSP, then his private key along with role data is sent to CSP by KM on a session basis.

User data access: The access is controlled at the CSP side itself by enforcing the access structure nr prior to sending the encrypted data.

Role and permission management: In the proposed system each user is attributed to a role and each role corresponding a specific permission set based on a access structure. Each and every user is categorized under a role set and each role set assigned with a set of permissions. The subjects or otherwise known as users granted with permissions through their roles may perform specific set of actions as allowed by the permission. Let us assume in general, a user u can be categorized under a role r whenever the user u was under a specific role r it will have a set of permissions 0. Unlike any other existing model the role r and their corresponding 0 are constrained with a time t, i.e., the roles and permission are subject to vary with time and they are session based. Its is inevitable that a data owner may have a set of users whose roles and their permissions to access the data objects may change time to time. This dynamic behavior of changing roles an privileges result in complexity of user and object management. We assume the proposed system state with reference to roles and permissions management as ", the proposed " can be defined over a collection of roles R. The users and permissions in the proposed system are u and 0, respectively when they are unconstrained by roles and permissions at the initial state.

A state S is a set of finite ordered listed elements (U, 0, UA, 0A) where U is a finite sub group corresponding U and 0 is finite subgroup belongs to set 0. The relation UA f U×R is concerned with the data user and their corresponding role assignment. Similarly, the role and their corresponding permissions are shown by the relation 0Ag0×R. The pair u and r belongs to the UA, i.e., (u, r) UA indicates that the particular subject u belongs to a role r. Likewise, the permissions and roles pair are also belongs to 0A and the relation among the pair is (p, r) PA. A user u is allowed with privileges 0 for an object if a role was there in the set rR such that (u, r)UA and (0, r) 0A. Overall in order to describe the working of the system, let us take the initial condition for the state "$_0$ = (U_0, 0_0, UA_0, $0A_0$).

Let as assume that h be the collection of command sets enforced for administration purpose and management of user, roles and their privileges. A two state of events are defined an initial state "$_0$ and " transition state ". Where as in the administration rule definition, one of the

state " = (U, 0, UA, 0A) and the transition level "$_0$ = (U_0, 0_0, UA_0, $0A_0$). A change from one level to another occurs, i.e., from " to "$_0$ only when an instruction 1, h (which can be described "1/6", "$_0$) when any one of the following conditions are satisfied.

AddU(u): This instruction executes addition of a user, the subject who wants to access data with privileges belongs to U (U, U_0 = Uc{u}, 0_0 = 0, UA_0 = UA and $0A_0$ = 0A.

DelU(u): This instruction executes deletion of user who already belongs to the user set, i.e., u belongs to U, U_0 = U{u}, P_0 = P, UA_0 = UA({(u, r), UA|r, R} and $0A_0$ = 0A.

AddObj(0): This instruction executes in order to add a new data object. The permission 0 belongs 0\0 such that U_0 = U, 0_0 = 0c{0}, UA_0 = UA and $0A_0$ = 0A.

DelObj(0): This instruction set deletes the data object 0 that belongs to 0, U_0 = U, 0_0 = 0({0} and $0A_0$ = 0A{(0, r), 0A|r, R.

AssignU(u, r): This instruction executes role r assignment to the user u whereas u and r belongs to the set U and R, respectively. U_0 = U, 0_0 = 0, UA_0 = UAc{(u, r)} and $0A_0$ = 0A (Algorithm 1).

Algorithm 1; Interaction of users, roles, permissions and their relation sets:

Roles (R):
r1-Admin, r2-physician, r3-doctor

Permissions (0):
01 = (read, medrecord)
02 = (write, medrecord)

Users (U):
u1-John, u2-whit, u3-heckm

UA relation (UA):
UA1(u1, r1)6UA1(john, admin)
UA2(u2, r2)6UA2(whit, physician)

0A relation (0A):
0A1-(01, r1)6(read, medrecord, admin)
0A2-(02, r1)6(write, medrecord, admin)

DesassignU(u, r): This instruction executes in order to de-assign a role r assigned to a user u. The user u and role r belongs to U and R, respectively, U_0 = U, 0_0 = 0, UA_0 = UA and $0A_0$ = 0A({(0, r)}.

GrantPer(0, r): This instruction grants permission 0 r. If 0 is permission and r is the role belong to 0, respectively U_0 = U, 0_0 = 0, UA_0 = UA and $0A_0$ = 0Ac{(0, r)}.

RevokePer(0, r): This instruction revokes permission 0 r. If 0 is permission and r is the role belong to 0, respectively U_0 = U, 0_0 = 0, UA_0 = UA and $0A_0$ = 0A({(0, r)}.

The phase of transition in the proposed system occurs as "0 h0/6", ..., h0n-1/6" normally under standard conditions.

Security policy architecture of the proposed system:

$$\forall u \in U \forall p \in P : Con(u, p) \Rightarrow \neg DoAccess(u, p)$$

The above equation represents the access control policy structure nr for architecture S having R roles and U users with P permissions. The Con(u, p) is the condition statement for the users u, U with permission set p, P. The overall working of the access control policy is defined by the condition statement Con(u, p).

The security structure requires isolation of roles and privileges, so that, a role assigned with a particular permission cannot be accessed by any other role. This role-permission control is exhibited by defining the proper roles and permissions as follow:

$$\forall u \in U \forall p \in P : \left[(u, r) \in UA \wedge (p, \tilde{r}) \in PA\right] \Rightarrow \neg DoAccess(u, p)$$

The role permission management between users also needs serious measures to check any cross data access problems. An user u belongs to a particular role should not be able to access his previous role permissions when he was assigned with a newer role:

$$\forall u \in U \forall p \in P : \left[(u, r) \in UA \wedge (p, \tilde{r}) \in PA\right] \vee \left[(u, \tilde{r}) \in UA \wedge (p, r) \in PA\right] \Rightarrow \neg DoAccess(u, p)$$

The proposed system provides for a robust data security provision in which a subject u may not be able to access the object unless he has sufficient permission to do so. This security can be described as:

$$Do\ access(u, p) = 0$$

when the subject not able to access the data.

Validation of security in the proposed scheme: In the following proof, we assume that a dishonest user i, trying to copy a genuine user's behavior u, the experiment is to stop the i by putting challenge code. Users, permissions, user-role assignment and permission-role assignment are chaosrepresented by U, P, UA and PA, respectively. The

relations among them are shown as $U \subseteq U$, $P \subseteq P$, $UA \subseteq U \times R$ and $PA \subseteq P \times R$. The corrupt users are represented by a set $Cr \subseteq U$ and the permissions in which the dishonest user to be challenged is represented by $Ch \subseteq P$. The experiments is given as follow as:

$$E^{ind}_{CBAC, x^{(\lambda)}}$$
$$b \leftarrow \$\{0,1\}; Cr, Ch \leftarrow 0$$
$$(st_M, FS, \{st[u]\} u \in U) \leftarrow \$ln i(1^{\lambda}, R)$$
$$b' \leftarrow \$x(1^{\lambda}, FS:0)$$
$$return (b' = b)$$

The proposed system is secure for all probable polynomial time conditions, we declare that:

$$Con^{ind}_{CBAC, x^{(\lambda)}} := \left| Pr\left[E^{ind}_{CBAC, x^{(\lambda)}} \rightarrow true \right] - \frac{1}{2} \right|$$

The detailed depiction of i's instruction sets and their execution is given as follow as.

UserUpdate:

For all $u \in U\{u^*\}$
$st[u] \leftarrow Update(st[u] msg_u)$
return (FS, msg_{u^*})

AddUser(U):

$U' \leftarrow U \cup \{u\}$
If (U', P, UA, PA) satisfies φ' then $U \leftarrow U'$; else return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AddUser(st_M, FS, u)$
return UserUpdates

DelUser(U):

$U' \leftarrow U / \{u\}; UA' \leftarrow (UA\{u\}) XR)$
If (U', P, UA', PA) satisfies φ' then $U \leftarrow U'$; $UA \leftarrow$ UAelse return $\perp (st_M, FS, \{msg_u\} u \in U) \leftarrow$ $\$DelUser(st_M, FS, u)$ return User Updates

AddObj(P):

$P' \leftarrow PU\{p\}$
If (U, P', UA, PA) satisfies φ' then $P \leftarrow P'$; $\leftarrow$ else return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AddObj(st_M, FS, p)$
return UserUpdates

DelObj(P):

$P' \leftarrow P / \{p\}; PA' \leftarrow PA / (\{p\}) XR)$
If (U, P', UA, PA') satisfies φ' then $P \leftarrow P'; PA \leftarrow$ PA'; elsereturn $\perp (st_M, FS\{msg_u\} u \in U) \leftarrow$ $\$DelObj(st_M, FS, p)$ return UserUpdates

AssignUser(u, r):

$UA' \leftarrow UA \cup \{(u, r)\}$
If (U, P, UA', PA) satisfies φ' then $UA \leftarrow U'$; UAelse return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AssignUser(st_M, FS, u, r)$
return UserUpdates

DeassignUser(u, r):

$UA' \leftarrow UA / \{(u, r)\}$
If (U, P, UA', PA) satisfies φ' then $UA \leftarrow U'$; else return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$DeassignUser(st_M, FS, u, r)$
return UserUpdates

GrantPer(p, r):

$P' \leftarrow PA \cup (\{p, r\})$
If (U, P, UA, PA') satisfies φ' then $P \leftarrow PA'; \leftarrow$ else return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$RevokePer(st_M, FS, p, r)$
return UserUpdates

RevokePer(p, r):

$PA' \leftarrow PA / (\{p, r\})$
If (U, P, UA, PA') satisfies φ' then $PA \leftarrow PA'; \leftarrow$ else return $\perp$
$(st_M, FS\{msg_u\} u \in U) \leftarrow \$RevokePer(st_M, FS, p, r)$
return UserUpdates

WriteP(p, m):

if $p \notin P$ then return $\perp$
$FS \leftarrow \$Write(FS, p, m)$
return FS

The malicious user i can use the instruction set to access the data content through the RBAC management system. Each instruction set is validated whether the user i able to access the data or not. The instruction sets able run their corresponding CBAC algorithm and update the proposed system. In this module the message $\{msg_u\} u, U$

is the output and at the end of each and every state the instruction set updates the condition of honest user. The instruction set sends message to the malicious user i.

The command set CORRUPTU enables the malicious user to access the behavior of a genuine user under attack, it also sends the data belongs to u to i, the user u is added to the list of corrupt users Cr.

The WRITE command implements a write instruction and the CHALLENGE implements a challenge in to the object further the code is added to the list Ch:

AddUser(U):

$$U \leftarrow U \cup \{u\}$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AddUser(st_M, FS, u)$$
$$\text{For all } u \in U / Cr$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

DelUser(U):

$$U \leftarrow U\{u\}; Cr \leftarrow Cr\{u\}; UA \leftarrow UA(\{u\} * R)$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$DelUser(st_M, FS, u)$$
$$\text{For all } u \in U/Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

AddObj (P):

$$P \leftarrow PU\{p\}$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AddObj(st_M, FS, p)$$
$$\text{For all } u \in U/Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

DelObj (P):

$$P \leftarrow PU\{p\}; Ch \leftarrow Ch / \{p\}; PA \leftarrow PA(p * R)$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$DelObj(st_M, FS, p)$$
$$\text{For all } u \in U/Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

AssignUser(u, r):

$$\text{if } u \in Cr \text{ and if for any } p \in Ch : (p, r) \in PA \text{ then return } \perp$$
$$UA \leftarrow UA / (\{u, r\})$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$AssignUser(st_M, FS, p)$$
$$\text{For all } u \in U / Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

DeassignUser(u, r):

$$UA \leftarrow UA / (\{u, r\})$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$DeassignUser(st_M, FS, p)$$
$$\text{For all } u \in U / Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

GrantPer(p, r):

$$\text{if } p \in Ch \text{ and if for any } u \in Cr : (u, r) \in UA \text{ then return } \perp$$
$$PA \leftarrow PA \cup \{(p, r)\}$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$GrantPer(st_M, FS, p, r)$$
$$\text{For all } u \in U / Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

chaosRevokePer(p, r):

$$PA \leftarrow PA \cup \{(p, r)\}$$
$$(st_M, FS\{msg_u\} u \in U) \leftarrow \$RevokePer(st_M, FS, p, r)$$
$$\text{For all } u \in U / Cr:$$
$$st[u] \leftarrow Update(st[u], msg_u)$$
$$\text{return } (FS\{msg_u\}) u \in Cr$$

CorruptU(U):

$$\text{if } p \notin U, \text{ then return } \perp$$
$$\text{for all } p \in Ch : \text{If } DoAccess(u, p) \text{ then return } \perp$$
$$Cr \leftarrow Cr \cup \{u\}; \text{return } st[u]$$

Write(p, m):

$$\text{if } p \notin P, \text{ then return } \perp$$
$$FS \leftarrow \$Write(FS, p, m)$$
$$\text{Return FS}$$

Challenge (p, m0, m1):

$$\begin{aligned}
&\text{if } p \notin P, \text{then return } \perp\\
&\text{for all } u \in Cr: \text{If DoAccess}(u,p) \text{ then return } \perp\\
&Ch \leftarrow Ch \cup \{p\}\\
&Fs \leftarrow \$\text{Write}(FS, p, m_b)\\
&\text{return } FS
\end{aligned}$$

U, Ch, the object will remain on the Ch as a malicious user even after the instruction set Write is called with p. This forbids access to the object since the user is listed as malicious user. If the access was not forbidden then the miscreant user can store the p before the WRITE instruction is executed. The READ command is not enabled for the test miscreant user as it may run the READ on its own with the FS and secret keys.

Security procedure implementation in the proposed system: In an access control system, it is essential for the envisioned security procedures adapted in the architecture to be implemented. An access structure denoted as nr can be defined by a conditional instruction Con, operating with a need of any one of the subject in the systems u and one of the data element p. If the given statement is satisfied then the instruction DoAccess (u, p). It means that the subject will not be able to reach the data object which is described as r, R:(u, r)óUAw(p, r)óPA. The proposed system executes the defined security procedure over the data object whenever the conditional instruction is Con (u, p) = 1, practically at this instant the subject will not be able to reach the necessary permissions p. Hence, forth we establish a security procedure to intervene whenever the Con instruction set is satisfied by the subject with necessary privileges p. So that, a malicious user cannot mimic a authenticated user and his corresponding permissions to decide between two choice of information be the content of permission p.

The ability of the proposed system to enforce the security procedure when a malicious user i try to imitate the authenticated user u in order to steal the content originally accessible to user u, is shown in the following analysis. i can try executing data access commands by imitating one of the authenticated users with his credentials. The proposed system must enforce the access control nr. The system is said to be failed to establish the security procedure when the i achieves Con (u, p) = 1 and the message is guessed by the i. If we assume that the probable value of the i to guess the data on p is 1/2 then it is established that the system executes the security procedures preventing data access to i. The proposed systems implements an access control nr as œu, Uœp, P: Con (u, p)Y¬DoAccess (u, p):

$$A_{CBAC,A}^{\text{ind-}\varphi'\text{-}(u,p)}(\lambda) := \left| \Pr\left[E_{CBAC,A}^{\text{ind-}\varphi'\text{-}(u,p)}(\lambda) \rightarrow \text{true} \right] - \frac{1}{2} \right|$$

When the user and privileges are of fixed value then the experiment is defined as follow as:

$$\begin{aligned}
&E_{CBAC,A}^{\text{ind-}\varphi'\text{-}(u,p)}(\lambda)\\
&b \leftarrow \$\{0,1\}\\
&(st_M, FS, \{st[u]\}u \in U) \leftarrow \$\ln i(1^\lambda, R)\\
&(m_0, m_1, st_\chi) \leftarrow \$\chi_1(1^\lambda, FS, st[u^*]: O)\\
&\text{if } con(u^*, p^*) = 0\\
&b' \leftarrow \$\{0,1\}; \text{return}(b' = b)\\
&FS \leftarrow \text{Write}(FS, p^*, m_b)\\
&b' \leftarrow \$\chi_2(st_\chi, FS)\\
&\text{return}(b' = b)
\end{aligned}$$

It is noteworthy that the imitating user has knowledge about the instruction commands needed to operate the system in order to obtain the data. The system implements an access control scheme nr, the system instruction commands that are against the policies as defined in the access structure nr are rejected by the system. Henceforth, every instruction is validated for consistency of the nr. When the access structure is not followed or violated it is returned with z. On other hand, when the instruction are in consistence with the access control policy nr, it regularly updates the state of the system in terms of (U, P, UA, PA) and also allows the algorithm corresponding to the particular instruction to follow. Output on the user end are displayed with genuine user information however msg_{u^*} is returned to the fake user.

Data erasing: There are two types of data erasing were performed, one is compulsory erasing and another one is optional erasing. Only the authorized data owner can perform this action. The DO sends that data identifier in its encrypted form to CSP. CSP then deletes the data corresponding to the file identifier. In case of compulsory erasure CSP validates the data stored in it by data users that are ancillary data allowed by DO. These user data and DO data are evaluated for malwares and are erased by CSP or quarantined. This compulsory erasure is to ensure data security flaws arising due to malware or cross scripts that may damage other data stored in the CSP.

RESULTS AND DISCUSSION

Performance analysis

Theoretical analysis: The proposed scheme is compared with some other existing access control schemes and evaluated in this study.

Key escrow issue: From the literature it is established that a number existing works have the key escrow problem chaoswhich is a major menace in cloud computing. A number of studies proposed to improve the security in cloud computing, however, failed to solve the key escrow problem. In this proposed scheme although, the Key is managed by a KM Module, it is a kept away from the data. By separating the data and keys on different CSPs, the KM may not able to achieve data access. Also KM selected in this proposed scheme is a trusted authority that is predominately a law enforcement agency owned cloud program. On other hands, if a DO not able to trust the KM, he may run his own KM Module which possibly very low online resources. This can hold good incase of an enterprise cloud client. Managing key on their own, by establishing their own cloud module a greater security is ensured solving the key escrow problem.

Apart from this, the CSP allows data access only through the enforced access structure. Hence, by having only role related data and key many not fruitful for the KM, if assumed it were being run on a untrusted CSP.

Brute force attack: Brute force attack is yet another type of problem in the cloud security. It usually happens duxring data transmission among different modules in the cloud system. Also when the hackers attack and steal data stored in CSP they may enable brute force attack to decipher the data. A semi-trusted honest but curious CSP can also try to read the data stored in its server by making a brute force attack. The brute force attack is about carrying out repeated trial and errors on the encrypted data with almost all possible combinations. In this proposed scheme the data during transmission as well when it was stored in the CSP, it is encrypted with key series generated by chaos theory. The key generated with key series known for their high degree of randomness. Making this chaos based encryption a strongest one against any brute force attacks. Due to the high degree of probability of key generated. It is will take infinite time to try all possible combinations. Hence, the proposed scheme provides an enhanced data security during storage and transmission.

Performance efficiency: The performance efficiency of number schemes has been compared with the proposed scheme. The comparison was carried out theoretically on storage and computational costs. The comparison is carried out on a simplified module of the proposed system excluding dynamic user management and any re-encryption involves in the process. Since, these two processes were already on the first time working of the scheme. Also, the transmission cost during the process has been omitted as it is only a negligible amount and found to be not a substantial parameter for improving the performance efficiency.

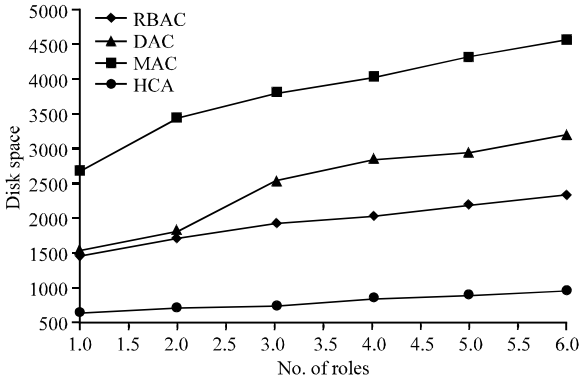

Fig. 3: Disk space requirements for various access control techniques

An access structure based control technique was used in combination with RBAC and chaos encryption. The access structure based access control is similar to that of a role based access control except the fact that roles permissions are defined through an access structure. Also, each data access is controlled not only based on the role assignment but also with an encryption technique. The send to the user is in the encrypted form and user alone can decipher and read it. Simple RBAC is widely used access control however it suffers from Trojan horse attack. In this research, data access is double secured with access control and mechanism hence any Trojan horse attack is nullified. Although, an attacker able to break the access structure, he will be stopped at the initial authentication level itself. Further, if he able to access any data by breaking the RBAC, the chaos based encryption embedded in the data secures the data. The attacker will not be able to access the data, since, the data is encrypted with highly random non predictable key serials generated with chaos based concept.

Figure 3 indicates the disk space requirements of various access control mechanism such as simple RBAC, DAC, MAC and HCA combination. It is important to note that in the proposed system occupies only a minimal disk space compared with other techniques. Unlike other techniques, in the proposed system only the access tree

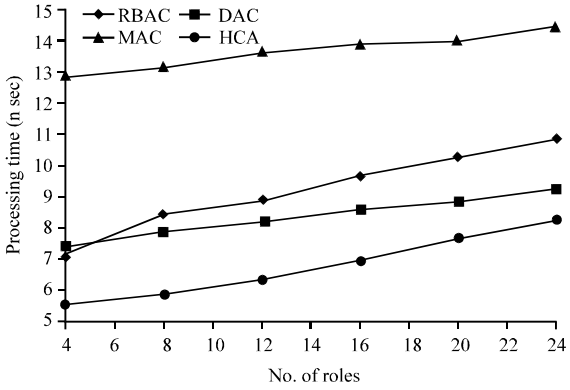

Fig. 4: Processing time and number of roles for different access control schemes

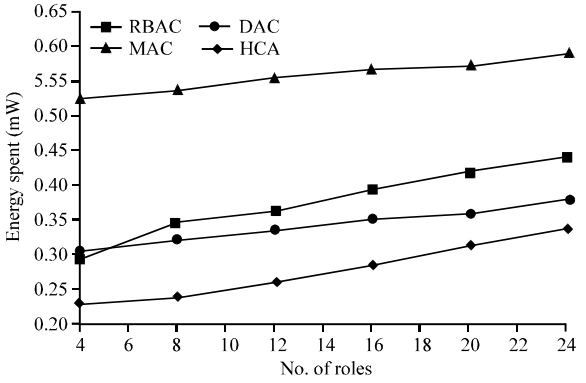

Fig. 6: Roles and energy expense comparison for various access control schemes

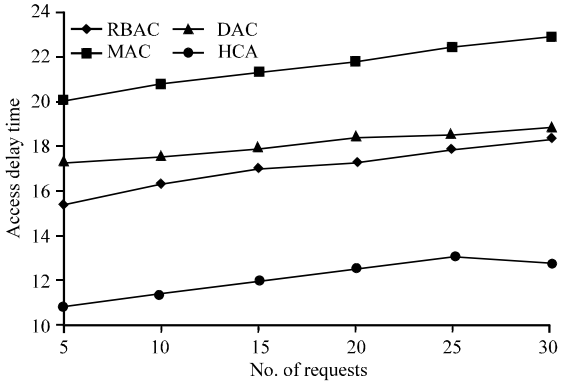

Fig. 5: Number of requests and access delay time analysis for various schemes

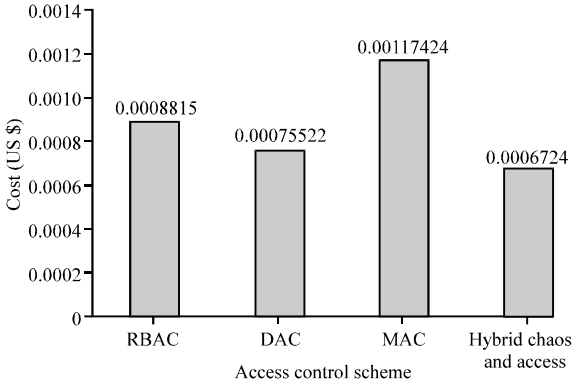

Fig. 7: Cost analysis for different access control schemes

is stored for each role and data is driven dynamically when the users tries to access it. Since, the data is already in encrypted form by chaos encryption, there is no need to separate them based on their roles fearing cross role attacks. This ensures greater access control along with data protection at a predominately low computational requirement.

Figure 4 illustrates the processing time and number of roles for different access control schemes such as RBAC, DAC, MAC and HCA. It is noted that the proportion of HCA is relatively low when compared with other schemes. However, in the proposed scheme while the number of roles increased, the processing time decreased.

Figure 5 illustrates the access delay time and number of request for different access control schemes such as RBAC, DAC, MAC and HCA. It is noted that the proportion of HCA is relatively low when compared with other schemes. However, in the proposed scheme while the number of requests increased, the access delay time decreased.

Figure 6 illustrates the number of roles and energy expense comparison for different access control schemes such as RBAC, DAC, MAC and HCA. It is noted that the energy expense of HCA is lower than other schemes. However, in the proposed scheme while the number of roles increased, the processing time decreased.

Figure 7 illustrates the cost analysis for different access control schemes such as RBAC, DAC, MAC and HCA. The cost is measured in USD. It is noted that the proportion of HCA is about 0.0006724. MAC shoots up to 0.00117424.

Figure 8 illustrates the access delay time for various access control schemes such as RBAC, DAC, MAC and HCA. The access delay time is measured in nano seconds (n sec). It is noted that the proportion of HCA is about 21.02. MAC shoots up to 37.23.

Figure 9 and 10 depict access delay time and processing time for varying access requests and number of roles. The proposed scheme manages larger number of users within a short time given in Fig. 11. Energy expenditure for various numbers of roles can be represented in Fig. 12.

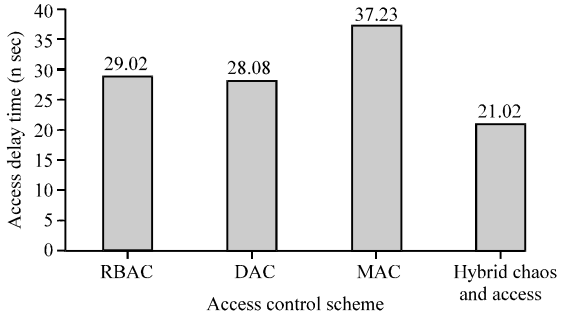

Fig. 8: Access delay time analysis for various access control schemes

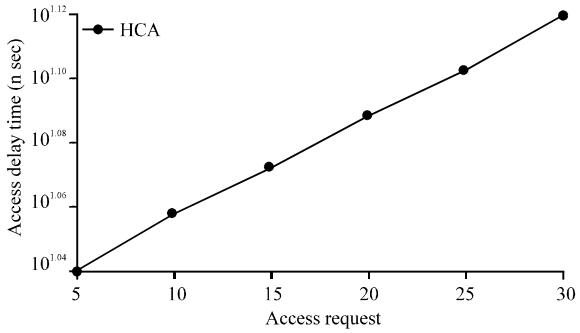

Fig. 9: Access requests and access delay time for the proposed HCA scheme

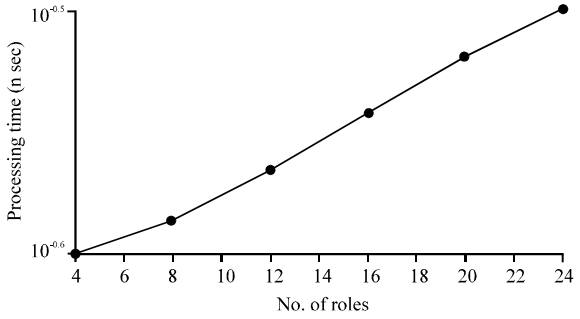

Fig. 10: Roles and processing time for HCA scheme

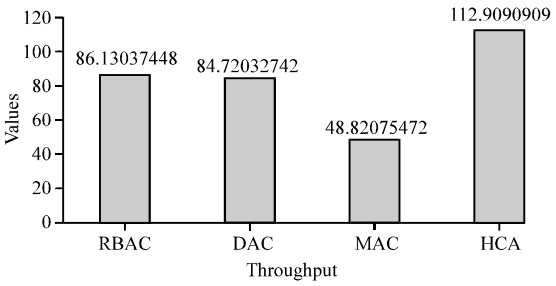

Fig. 11: Access control throughput comparison for various schemes

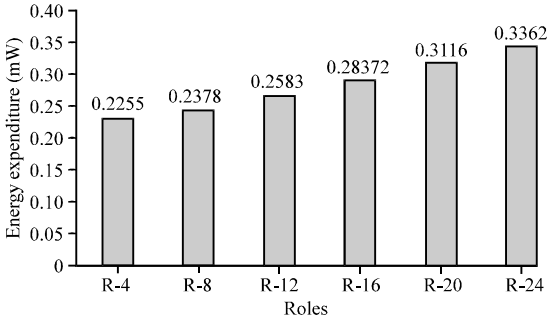

Fig. 12: Roles and energy relationship for HCA

CONCLUSION

A novel approach for data security and privacy has been proposed using three layered architecture. The frame work is a combination of RBAC and chaos (HRCA).This work allows only authorized users to access sensitive data. Moreover, the data should be available to legitimate users without any denial of service using break-the-glass approach. Although, the key manager is aware of the keys, it is not aware of the access structure defined, which is located in the cloud server that avoids any possible attack. The anonymization module preserve privacy of data collected using valuable resources of the country. This dynamic access control is more suitable for dynamic environment with instant modernizes of resources.

REFERENCES

Ali, K.H., D. Greenwood, J.W. Smith and I. Sommerville, 2012. The cloud adoption toolkit: Supporting cloud adoption decisions in the enterprise. Software Pract. Experience, 42: 447-465.

Armbrust, M., A. Fox, R. Griffith, A.D. Joseph and R. Katz *et al*., 2010. A view of cloud computing. Commun. ACM, 53: 50-58.

Bethencourt, J., A. Sahai and B. Waters, 2007. Ciphertext-policy attribute-based encryption. Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07.), May 20-23, IEEE, New York, USA., ISBN: 0-7695-2848-1, pp: 321-334.

Damiani, M.L, E. Bertino, B. Catania and P. Perlasca, 2007. GEO-RBAC: A spatially aware RBAC. ACM Transa. Inform. Syst. Secur., 10: 1-42.

Dawn, S., E. Shi, I. Fischer and U. Shankar, 2012. Cloud data protection for the masses. Comput., 45: 39-45.

DiVimercati, S.D.C., S. Foresti, S. Jajodia, S. Paraboschi and P. Samarati, 2007. Over-encryption: Management of access control evolution on outsourced data. Proceedings of the 33rd International Conference on Very large Data Bases, September 23-28, 2007, VLDB Endowment Inc., Austria, ISBN: 978-1-59593-649-3, pp: 123-134.

Foster, I., Y. Zhao, I. Raicu and S. Lu, 2008. Cloud computing and grid computing 360-degree compared. Proceedings of the Workshop on Grid Computing Environments GCE'08, November 12-16, 2008, IEEE, Austin, Texas, USA, ISBN:978-1-4244-2860-1, pp: 1-10.

Hariri, S., P. Varshney, L. Zhou, V.V. Menon and S. Ghaya, 1999. A hierarchical analysis approach for high performance computing and communication applications. Proceedings of the 32nd Hawaii International Conference on System Sciences, January 1, 1999, IEEE Computer Society Washington, DC, USA., pp: 1-8.

Hawang, K. and D. Li, 2010. Trusted cloud computing with secure resources and data coloring. IEEE Comput. Soc., 14: 14-22.

Marston, S., Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, 2011. Cloud computing: The business perspective. Decis. Supp. Syst., 51: 176-189.

Osborn, S., R. Sandhu and Q. Munawer, 2000. Configuring role-based access control to enforce mandatory and discretionary access control policies. ACM Trans. Inform. Syst. Secur., 3: 85-106.

Rost, P., C.J. Bernardos, A.D. Domenico, M.D. Girolamo and M. Lalam *et al*., 2014. Cloud technologies for flexible 5G radio access networks. IEEE. Commun. Mag., 52: 68-76.

Sekhar, B.R., B.S. Kumar, L.S. Reddy and V.P. Chandar, 2012. CP-ABE based encryption for secured cloud storage access. Intl. J. Sci. Eng. Res., 3: 1-5.

Victor, E., L.M. Liebrock and D. Shin, 2010. Permission management system: Permission as a service in cloud computing. Proceedings of the IEEE 34th Annual Conference on Computer Software and Applications Conference Workshops (COMPSACW), July 19-23, 2010, IEEE, Seoul, Korea, ISBN: 978-1-4244-8089-0, 371-375.

Zhou, L., V. Varadharajan and M. Hitchens, 2011. Enforcing role-based access control for secure data storage in the cloud. Comput. J., 54: 1675-1687.

Zhou, M., R. Zhang, W. Xie, W. Qian and A. Zhou, 2010. Security and privacy in cloud computing: A survey. Proceedings of the 6th IEEE International Conference on Semantics Knowledge and Grid (SKG), November 1-3, 2010, IEEE, Beijing, China, ISBN: 978-1-4244-8125-5, pp: 105-112.

Zomaya, A.Y. and Y.C. Lee, 2012. Energy Efficient Distributed Computing Systems. Vol. 88, John Wiley & Sons, Hoboken, New Jersey.