Secure Role-Based Access Control on Encrypted Data in Cloud Storage using ARM

Rohini Vidhate, V. D. Shinde

Abstract— **With the rapid developments occurring in cloud computing and services, there has been a growing trend to use the cloud for large-scale data storage. This has raised the important security issue of how to control and prevent unauthorized access to data stored in the cloud. One well known access control model is the role-based access control (RBAC), which provides flexible controls and management by having two mappings, users to roles and roles to privileges on data objects. In this paper, we propose a role-based encryption (RBE) scheme that integrates the cryptographic techniques with RBAC. Our RBE scheme allows RBAC policies to be enforced for the encrypted data stored in public clouds. Based on the proposed scheme, we present a secure RBE-based hybrid cloud storage architecture that allows an organization to store data securely in a public cloud, while maintaining the sensitive information related to the organization's structure in a private cloud. We describe a practical implementation of the proposed RBE-based architecture and discuss the performance results. We demonstrate that users only need to keep a single key for decryption, and system operations are efficient regardless of the complexity of the role hierarchy and user membership in the system. AES encryption/decryption algorithm is used for key based encryption. We have used trusted computing endorsement key approach for implementing Key based encryption/decryption algorithm on ARM. This approach made the system more secure and reliable.**

Index Term- Cloud computing, Encryption/Decryption, AES Algorithm

I. INTRODUCTION

THERE has been a growing trend in the recent times to store data in the cloud with the dramatic increase in the amount of digital information such as consumers' personal data to larger enterprises wanting to back up databases or store archival data. Cloud data storage can be particularly attractive for users (individuals or enterprises) with unpredictable storage demands, requiring an inexpensive storage tier or a low cost, long-term archive. By outsourcing users' data to the cloud, service providers can focus more on the design of functions to improve user experience of their services without worrying about resources to store the growing amount of data. Cloud can also provide on demand resources for storage which can help service providers to reduce their maintenance costs. Furthermore, cloud storage can provide a flexible and convenient way for users to access their data from anywhere on any device.

In role-based access control (RBAC) model, roles are mapped to access permissions and users are mapped to appropriate roles. For instance, users are assigned membership to the roles based on their responsibilities and qualifications in the organization. Permissions are assigned to qualified roles instead of individual users. Moreover, in RBAC, a role can inherit permissions from other roles, hence there is a hierarchical structure of roles. Since being first formalized in 1990's, RBAC has been widely used in many systems to provide users with flexible access control management, as it allows access control to be managed at a level that corresponds closely to the organization's policy and structure

In traditional access control systems, enforcement is carried out by trusted parties which are usually the service providers. In a public cloud, as data can be stored in distributed data centres, there may not be a single central authority which controls all the data centres. Furthermore the administrators of the cloud provider themselves would be able to access the data if it is stored in plain format. To protect the privacy of the data, data owners employ cryptographic techniques to encrypt the data in such a way that only users who are allowed to access the data as specified by the access policies will be able to do so. We refer to this approach as a policy based encrypted data access. The authorized users who satisfy the access policies will be able to decrypt the data using their private key, and no one else will be able to reveal the data content. Therefore, the problem of managing access to data stored in the cloud is transformed into the problem of management of keys which in turn is determined by the access policies.

In this paper, we present the design of a secure RBAC based cloud storage system where the access control policies are enforced by a new role-based encryption (RBE) that we proposed in the paper. This RBE scheme enforces RBAC policies on encrypted data stored in the cloud with an efficient user revocation using a broadcast encryption1 mechanism described in [3]. In our RBE scheme, the owner of the data encrypts the data in such a way that only the users with appropriate roles as specified by a RBAC policy can decrypt and view the data. The role grants permissions to users who qualify the role and can also revoke the permissions from existing users of the role. The cloud provider (who stores the data) will not be able to see the content of the data if the provider is not given the appropriate role. Our RBE scheme is able to deal with role hierarchies, whereby roles inherit permissions form other roles. A user is able to join a role after the owner has encrypted the data for that role. The user will be able to access that data from then on, and the owner does not need to re-encrypt the data. A user can be revoked at any time in which case, the revoked user will not have access to any future encrypted data for this role. With our new RBE scheme, revocation of a user from a role does not affect other users and roles in the system. In addition, we outsource part of the decryption computation in the scheme to the cloud, in which only public parameters are involved. By using this approach, our RBE scheme achieves an efficient decryption on the client side. We have also used the same strategy of outsourcing to improve the efficiency of the management of user to role memberships, involving only public parameters.

A. *Overview of the project*

Based on the proposed RBE scheme, we develop a secure cloud data storage architecture using a hybrid cloud infrastructure. This hybrid cloud architecture is a composite of private cloud and public cloud, where the private cloud is used to store only the organization's sensitive structure information such as the role hierarchy and user membership information, and the public cloud is used to store the actual data that is in the encrypted form

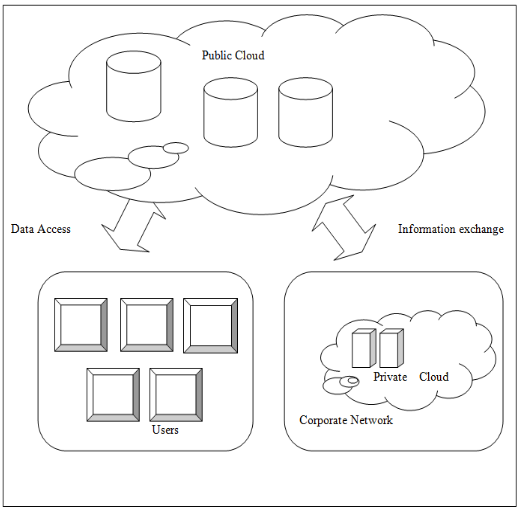


Fig 1 Hybrid Cloud Storage

B. *Related Work*

One approach to enforce access control policies is to transform the access control problem into a key management problem. In the literature, there exist many hierarchy access control schemes [1] which have been constructed based on hierarchical key management (HKM) schemes, and approaches using HKM schemes to enforce RBAC policies for data storage are discussed in [1]. However, these solutions have several limitations. For instance, if there is a large number of data owners and users involved, the overhead involved in setting up the key infrastructure can be very high indeed. Furthermore, when a user's access permission is revoked, all the keys known to this user as well as all the public values related to these keys need to changed, which makes these schemes impractical. An alternative approach for the management of keys is Hierarchical ID-based Encryption (HIBE), such as [1].However, in a HIBE scheme, the length of the identity becomes longer with the growth in the depth of hierarchy. In addition, the identity of a node must be a subset of its ancestor node so that its ancestor node can derive this node's private key for decryption. Therefore, this node cannot be assigned as a descendant node of another node in the hierarchy tree unless the identity of the other role is also the super set of this node's identity. Recently we have seen the development of schemes built directly on RBAC policies. We introduced a role-based encryption scheme (RBE) in [1]. However, the user revocation in this scheme requires the update of all the role related parameters. Another scheme was proposed in [1]. In this scheme, the size of the cipher text increases linearly with the number of all the predecessor roles. In addition, if a user belongs to different roles, multiple keys need to be possessed by this user. Moreover, the management of the user membership for each individual role requires the use of the system secret keys. The scheme proposed in this paper overcomes these limitations, and each role can use its own secret keys to manage the user membership without the need to know the system secret keys.

Moreover, the scheme proposed in this paper provides efficient user revocation. Besides RBAC, there are also other access control models such as Attribute Based Access Control (ABAC). In ABAC, access is granted based on attributes of the user. Systems define combination of attributes as the access policies, and users need to prove that they have these attributes in order to gain access. In 2006, the first attribute-based encryption (ABE) scheme was proposed in [5] based on the work in [1], and some other ABE schemes have been proposed afterwards. In these schemes, data is encrypted to a set of attributes, and users who have the private keys associated with these attributes can decrypt the data. These works have provided an alternative approach to secure the data stored in a distributed environment using a different access control mechanism, in [1]; we have shown that an ABE scheme can be used to enforce RBAC policies. However, in that approach, the size of user key is not constant, and the revocation of a user will result in a key update of all the other users of the same role. [1] Also investigated the solutions of using ABE scheme in RBAC model. However their solution only maps the attributes to the role level in RBAC, and they assumed that the RBAC system itself would determine the user membership. Other approaches to protect data privacy in a cloud environment include using direct encryption and proxy re encryption. In these cryptographic schemes, data is allowed to be encrypted directly to the users with whom the owners wish to share the data. This is analogous to the access control policies in Discretionary Access Control (DAC) model. Hence they are usually used in systems where DAC model is adopted. Since the permissions in such systems are specified either in a flat out structure or in an access matrix, we do not compare them with our schemes as the access policies are specified differently in RBAC model.

C. *Premilinaries*

Our RBE scheme has the following four types of entities. SA is a system administrator that has the authority to generate the keys for users and roles, and to define the role hierarchy. RM is a role manager2 who manages the user membership of a role. Owners are the parties who want to store their data securely in the cloud. Users are the parties who want to access and decrypt the stored data in the cloud. Cloud is the place where data is stored and it provides interfaces so all the other entities can interact with it. We define the following algorithms for our RBE scheme:

Setup (λ) takes as input the security parameter λ and outputs a master secret key mk and a system public key pk.

mk is kept secret by the SA while pk is made public to all users of the system.

Extract (mk, ID) is executed by the SA to generate the key associated with the identity ID. If ID is the identity of a user, the generated key is returned to the user as the decryption key. If ID is the identity of a role, the generated key is returned to the RM as the secret key of the role, and an empty user list RUL which will list all the users who are the members of that role is also returned to the RM.

ManageRole (mk, IDR, PRR) is executed by the SA to manage a role with the identity IDR in the role hierarchy. PRR is the set of roles which will be the ancestor roles of the role. This operation publishes a set of public parameters pubR to cloud.

AddUser (pk, skR, RULR, IDU) is executed by the role manager RM of a role R to grant the role membership to the user IDU, which results in the role public parameters pubR and role user list RULR, being updated in cloud.

RevokeUser (pk, skR, RULR, IDU) is executed by a role manager RM of a role R to revoke the role membership from a user IDU , which also results in the role public parameters pubR and role user list RULR, being updated in cloud.

Encrypt (pk, pubR) is executed by the owner of a message M. This algorithm takes as input the system public key pk, the role public parameters pubR, and outputs a tuple When this operation finishes, a cipher text is output and uploaded to cloud by the owner.

Decrypt (pk, pubR, dk, C) is executed by a user who is a member of the role R. This algorithm takes as input the system public key pk, the role public parameters pubR, the user decryption key dk, the part C from the cipher text downloaded from cloud, and outputs the message encryption key $K \in k$.

II. IMPLEMENTATION OF SYSTEM

A. Architecture

In this section, we present the architecture of our secure cloud storage system. It is a hybrid cloud architecture comprising a private cloud which is used to store sensitive role hierarchy of the organization and user memberships, and a public cloud storing the encrypted data and public parameters associated with the RBE system. The users who wish to access the encrypted data and the data owners who wish to encrypt their data only interact with the public cloud. The role hierarchy and user to role mappings related to the organization are maintained in the private cloud which is only accessible to the administrator of the organization. The administrator specifies the role hierarchy and the role managers who manage the user membership relations.

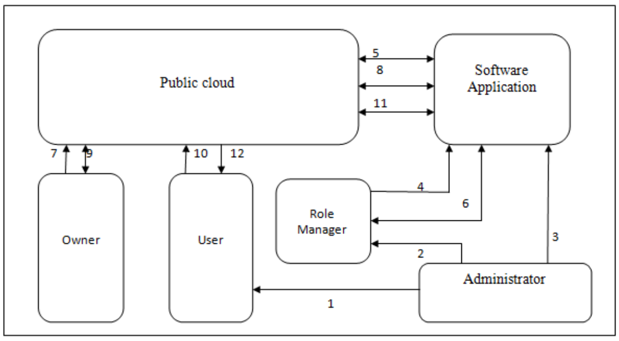
We first consider the components of the system architecture shown in Figure 2.

Public Cloud

The numbers shown in the figure refer to the system operations which will be described in Section V-B. Public cloud is a third party cloud provider which resides outside the infrastructure of the organizations, and organizations outsource their users' encrypted data to the public cloud. Since the public cloud is untrusted, data stored in the public cloud could be accessed by unauthorized parties, such as employees of the cloud provider and users from other organizations who are also using services from the same cloud. Therefore only public information and encrypted data will be stored in the public cloud. An untrusted public cloud may deny a user's request for accessing stored data in the cloud or provide users within correct data. Such behaviors will result in the users not being able to access the data stored in cloud (cf denial of service attacks), but will not cause violation of RBAC policies. These behaviors can be detected, as a user can observe the failure immediately after s/he communicates with the public cloud. In this case, organization may choose to change the cloud provider to a more reliable one, especially if the current provider is found to be malicious. The discussion of such denial of service attacks type behaviors is beyond the scope of this paper; hence in this paper, we will assume that the public cloud will faithfully execute the steps of the proposed RBE scheme and provide valid responses to users' requests.

User

Users are the parties who wish to acquire certain data from the public cloud. Each user is authenticated by the administrator of the role-based system; upon successful authentication, the user is given a secret key which is associated with the identity of the user. (In this paper, though we do not address the authentication, any one of suitable authentication schemes can be used for this purpose). Users are not involved in any process related to organization structure updates, including user membership updates and changes in the role hierarchy. So they are not allowed to communicate directly with the private cloud.



Role Manager

A role manager is the party who manages the relationship between users and roles. Each role has its own role parameters which defines the user membership. These roles parameters are stored in the private cloud. When updating the user membership of a role, the role manager needs to compute new role parameters and update them in the private cloud. None of users are affected by this operation, so role managers do not need to communicate with users, and they only need to interact with the private cloud. Before a user is included into a role, the role manager will need to authenticate the user in order to ensure that the user qualifies for the role. We do not consider the authentication mechanisms in this paper; we assume that such mechanisms exist and role managers will grant role membership only to appropriately qualified users in the system.

Administrator

The administrator is the certificate authority of the organization. The administrator generates the system parameters and issues all the necessary credentials. In addition, the administrator manages the role hierarchy structure of the organization. To put a role into the organization's hierarchy structure, the administrator computes the parameters for that role. These parameters represent the position of the role in the role hierarchy, and are stored in the private cloud. When the role hierarchy changes, the administrator updates these parameters for the roles that have been changed in the private cloud.

Owner

Owners are the parties who possess the data and want to store the encrypted data in the public cloud for other users to access; owners specify who can access the data in terms of role-based policies. In the RBAC model, they are the parties who manage the relationship between permissions and roles. An owner can be a user within the organization or an external party who wants to send data to users in the organization. In this architecture, we consider an owner to be a logically separate component even though a user can be an owner and vice versa. Owners only interact with the public cloud, and no secret values are required for these interactions. They do not have to keep any parameters in the RBE scheme, and they need to obtain all the required parameters from the public cloud when they perform their encryption operations.

B. System Operations

Now we describe the system operations of our proposed architecture using the steps shown in figure 3. Assume the system uses a secure encryption scheme Enc to encrypt messages using the key generated in the Encrypt algorithm.

Extract

This operation is executed by the administrator to add a user or a role into the system. Step 1 represents the interaction to generate a decryption key for a user, and step 2represents the interaction to generate a role secret for a role. The administrator computes the secret dk for the user or sk for the role, and sends the secret to the user or role via a secret channel;

ManageRole

The operation of managing a role in the role hierarchy structure is also executed by the administrator. Steps3 represents the interactions in the management of a role. The administrator decides the inheritance relationship of the role, and updates the position of a role in the role hierarchy structure. This is done in step 3 where the administrator computes and uploads the following tuple to the private cloud, $<IDR, AR, BR, PR>$ where IDR is the identity of the role, AR, BR are computed as shown in the ManageRole algorithm, and PR is the set of all the immediate roles that inherit permissions from the role IDR.

Add User/Revoke User

These two operations are performed by role managers to update the user membership of roles, and the interactions for these operations are represented by steps4 to 6. When adding or revoking users, a role manager sends the pair $< IDU , \tau >$ (IDU is the user identity and τ indicates the type of operation - add or revoke) to the private cloud in step4. In step 5, the private cloud forwards this request to the public cloud and the public cloud computes and returns YR to the private cloud. In step 6, the private cloud forwards the value YR to the role manager, and the role manager verifies YR, computes and uploads the following tuple to the private cloud, $< IDU , TR, WR, VR, SR >$ where IDU is the identity of the user, and TR, WR, VR, SR are the values computed as in AddUser / RevokeUser algorithms.

Encryption

Steps 7 to 9 show the processes involved in encrypting a message. When an owner wants to encrypt data M to a role R, s/he retrieves the role public parameters from the public cloud as part of the encryption key, which is shown as step 7. Since these parameters are stored in the private cloud, the public cloud forwards the owner's request to the private cloud, and the private cloud passes back the following tuple to the public cloud in step 8. In step 9, the public cloud simply forwards the tuple to the owner. Upon receiving the role public parameters, the owner checks if the timestamp t is up-to-date and verifies the attached signature to check its validity and whether it is issued by the private cloud. If the role public parameters are verified to be valid, then the owner computes and uploads the following cipher text to the public cloud, $< C1, C2, C3, Enck\ (M) >$ After receiving the cipher text, the public cloud generates a unique index to identify the message, and stores this index value pair in the cloud.

Decryption

Steps 10 and 12 show the processes involved in data decryption. When a user U wants to view the data M that has been previously encrypted and stored in the public cloud, the user first requests the cipher text of M from the public cloud in step 10. Since the role parameters used in decryption are stored in the private cloud, the public cloud needs to request these parameters from the private cloud.

III. PROPOSED ARCHITECTURE BLOCK DIAGRAM DESCRIPTION

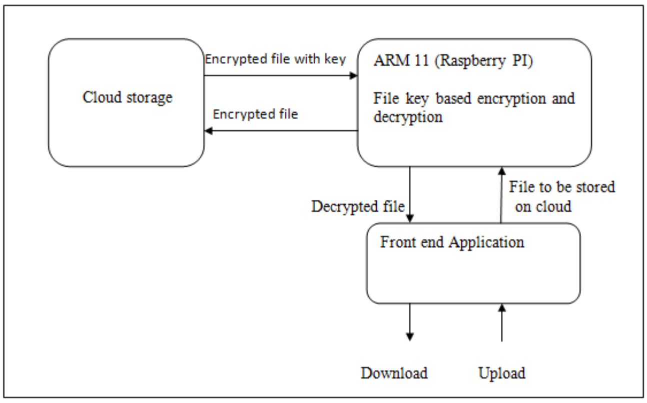


Fig. 3 Proposed system architecture

Front end Application

Main function of this application is implementing RBAC model. All user and roles are created in this model. Users are assigned with a single role or multiple roles. Authentication of user is also done in this model itself. This application will be created in Java.

ARM 11 Processor

The algorithm to encrypt or decrypt the files will be implemented on ARM. ARM processor encrypts and decrypts the file uploaded by the user. While encrypting the files a special key will be generated. This generated key is associated with the encrypted file.

Whenever user wants to download the content, authorization and authentication is done by front end application. After successful authentication arm matches the user provided key with the one associated with the encrypted file. When the match is perfect then only the file is decrypted and user can download the file.

Encryption, Decryption, Key generation and Key association with file, Key matching will be done in arm processor.

AES Algorithm

AES algorithm is used for encryption and decryption. Please refer figure 4 for AES Algorithm for encryption and decryption.

Cloud (Storage space)

Files are physically stored on cloud. This is secure cloud and the reason is that the file stored here are in encrypted format and Cloud service provider also cannot decrypt it as it is key protected.

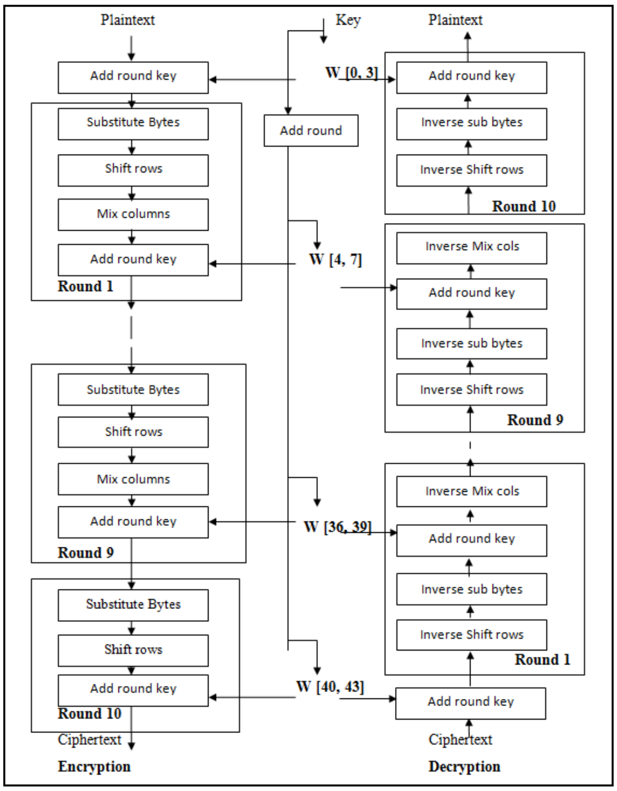


Fig. 4 Flow Chart for Encryption / Decryption algorithm

IV. CONCLUSION

In this paper, first we proposed a new RBE scheme that achieves efficient user revocation. Then we presented a RBAC based cloud storage architecture which allows an organization to store data securely in a public cloud, while maintaining the sensitive information related to the organization's structure in a private cloud. Then we have developed a secure cloud storage system architecture and have shown that the system has several superior characteristics such as constant size cipher text and decryption key. From our experiments, we observe that both encryption and decryption computations are efficient on the client side, and decryption time at the cloud can be reduced by having multiple processors, which is common in a cloud environment. We believe that the proposed system has the potential to be useful in commercial situations as it captures practical access policies based on roles in a flexible manner and provides secure data storage in the cloud enforcing these access policies.

REFERENCES

[1] Lan Zhou, Vijay Varadharajan, and Michael Hitchens "Secure Role-Based Access Control on Encrypted Data in Cloud Storage." IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, 12, DECEMBER 2013.

[2] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in Proc. IEEE INFOCOM, Mar. 2010, pp. 534–542..

[3] Y. Zhu, D. Ma, C. Hu, and D. Huang, "How to use attribute-based encryption to implement role-based access control in the cloud," in Proc. Int. Workshop Sec. Cloud Comput., 2013, pp. 33–40.

[4] Y. Zhu, D. Ma, C. Hu, and D. Huang, "How to use attribute-based encryption to implement role-based access control in the cloud," in Proc. Int. Workshop Sec. Cloud Comput., 2013, pp. 33–40.

[5] "Efficient implementation of Advanced Encryption Standard (AES) for ARM based platforms" IEEE Conference paper 17 March 2012.

[6] Yang Jian-wei, Yang Jian-xiang. Linux Transplant Based on the Processor of S3C2410 [J]. Information Technology, 2007, (8):0097-0100.

[7] Global Survey: Has Cloud Computing Matured? [Online]. Available: http://www.avanade.com/Documents/Research%20and%20Insights/Global_Survey_Slide_Graphics_Has_Cloud_ Matured.pdf.

[8] Yakun Liu, Xiaodong Cheng,"Design and Implementation of Embedded Web Server Based on ARM and Linux " In: 2010 2nd International Conference on Industrial Mechatronics and Automation.

[9] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage,"in Proc. NDSS, Feb. 2005, pp. 29–43.