ACCOMPLISHING FLEXIBLE AND SELF-CONTAINED INFORMATION PROTECTION IN CLOUD COMPUTING

D. Prem Kumar, Assistant Professor ,S V C E
K Sree Divya , Assistant Professor,SVCE
T Vijaya Rao, Assistant Professor, SVCE
S Swarnalata, Assistant Professor, SVCE

Abstract:
For enterprise systems running on public clouds in which the servers are outside the control domain of the enterprise, access control that was traditionally executed by reference monitors deployed on the system servers can no longer be trusted. Hence, a self-contained security scheme is regarded as an effective way for protecting outsourced data. However, building such a scheme that can implement the access control policy of the enterprise has become an important challenge. In this paper, we propose a self-contained data protection mechanism called RBAC-CPABE by integrating role-based access control (RBAC), which is widely employed in enterprise systems, with the ciphertext-policy attribute-based encryption (CP-ABE). First, we present a data-centric RBAC (DC-RBAC) model that supports the specification of fine-grained access policy for each data object to enhance RBAC's access control capabilities. Then, we fuse DC-RBAC and CP-ABE by expressing DC-RBAC policies with the CP-ABE access tree and encrypt data using CP-ABE. Because CP-ABE enforces both access control and decryption, access authorization can be achieved by the data itself. A security analysis and experimental results indicate that RBAC-CPABE maintains the security and efficiency properties of the CP-ABE scheme on which it is based, but substantially improves the access control capability. Finally, we present an implemented framework for RBAC-CPABE to protect privacy and enforce access control for data stored in the cloud.

Index Terms—Role-based access control, ciphertext-policy attribute-based encryption, self-contained data protection, cloud computing

INTRODUCTION

In many situations, when a user encrypts sensitive data, it is imperative that she establish a specific access control policy on who can decrypt this data. For example, suppose that the FBI public corruption offices in Knoxville and San Francisco are investigating an allegation of bribery involving a San Francisco lobbyist and a Tennessee congressman. The head FBI agent may want to encrypt a sensitive memo so that only personnel that have certain credentials or at tributes can access it. For instance, the head agent may specify the following access structure for accessing this information: ((“Public Corruption Office” AND (“Knoxville” OR “San Francisco”)) OR (management-level > 5) OR “Name: Charlie Eppes”). By this, the head agent could mean that the memo should only be seen by agents who work at the public corruption offices at Knoxville or San Francisco, FBI officials very high up in the management chain, and a consultant named Charlie Eppes. As illustrated by this example, it can be crucial that the person in possession of the secret data be able to choose an access policy based on specific knowledge of the underlying data. Furthermore, this person may not know the exact identities of all other people who should be able to access the data, but rather she may only have a way to describe them in terms of descriptive attributes or credentials. Traditionally, this type of expressive access control is enforced by employing a trusted server to store data locally. The server is entrusted as a reference monitor that checks that a user presents proper certification before allowing him to access records or files. However, services are increasingly storing data in a distributed fashion across many servers. Replicating data across several locations has advantages in both performance and reliability. The drawback of this trend is that it

is increasingly difficult to guarantee the security of data using traditional methods; when data is stored at several locations, the chances that one of them has been compromised increases dramatically. For these reasons we would like to require that sensitive data is stored in an encrypted form so that it will remain private even if a server is compromised.

In RBAC, access permissions are assigned through rolesand cannot be directly assigned to a user, which is insufficiently fine-grained. For example, suppose that user *ux* needs to be granted permission *p*. In the RBAC model there are two ways to achieve this goal. The first approach is to assign the permission *p* to one of *ux*'s roles *r*. However, it means that all users who are assigned to role *r* are also granted permission *p*, which may introduce security problems. The second approach is to add a new role *r*' and assign it to *ux*. Although this approach solves the problem raised by the first approach, adding an additional role *r*' increases the complexity of the system—especially when such authorizations are very frequent. Thus, neither approach can effectively achieve the goal.

• *RBAC describes an access control policy for the full collection of data in the entire enterprise rather than for each data object.* By defining roles and assigning those roles to users, RBAC can achieve data protection. However, data is only one constituent of a system (i.e. users, roles, permission assignments and so forth can have constraints, but data cannot). Hence, RBAC is targeted mainly to integral control of the data in the system, but it cannot meet the specific security requirements of each data object.

• *RBAC needs to be implemented using reference monitors that run on the data servers.* Because cloud servers may not always be trusted, depending on them to enforce access control introduces insecurities into the system.

RELATED WORK

In this work, we provide the first construction of a ciphertext-policy attribute-based encryption (CP-ABE) to address this problem, and give the first construction of such a scheme. In our system, a user's private key will be associated with an arbitrary number of attributes expressed as strings. On the other hand, when a party encrypts a message in our system, they specify an associated access structure over attributes. A user will only be able to decrypt a ciphertext if that user's attributes pass through the ciphertext's access structure. At a mathematical level, access structures in our system are described by a monotonic "access tree", where nodes of the access structure are composed of threshold gates and the leaves describe attributes. We note that AND gates can be constructed as n-of-n threshold gates and OR gates as 1-of-n threshold gates. Furthermore, we can handle more complex access controls such as numeric ranges by converting them to small access trees (see discussion in the implementation section for more details).

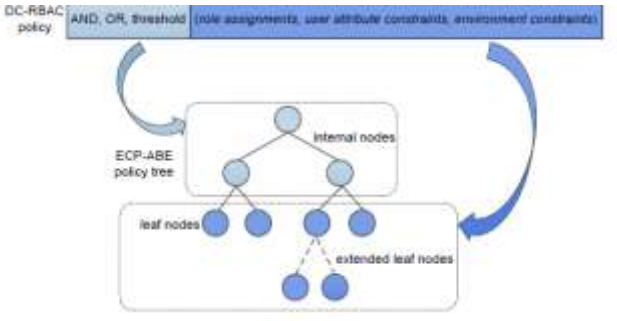
Collusion Resistance and Attribute-Based Encryption The defining property of Attribute-Based Encryption systems are their resistance to collusion attacks. This property is critical for building cryptographic access control systems; otherwise, it is impossible to guarantee that a system will exhibit the desired security properties as there will exist devastating attacks from an attacker that manages to get a hold of a few private keys. While we might consider ABE systems with different flavors of expressibility, prior work [4, 15] made it clear that collusion resistance is a required property of any ABE system.

Before attribute-based encryption was introduced there were other systems that attempted to address access control of encrypted data [9, 8] by using secret sharing schemes [12, 9, 6, 5, 3] combined with identity-based encryption; however, these schemes did not address resistance to collusion attacks. Recently, Kapadia, Tsang, and Smith [9] gave a cryptographic access control scheme that employed proxy servers. Their work explored new methods for employing proxy servers to hide policies and use non-monontonic access control for small universes of attributes. We note that although they called this scheme a form of CP-ABE, the scheme

does not have the property of collusion resistance. As such, we believe that their work should not be considered in the class of attribute-based encryption systems due to its lack of security against collusion attacks.

Expressing DC-RBAC policy with ECP-ABE

To construct RBAC-CPABE, two problems must be solved. The first problem involves how to support role assignment in ECP-ABE. Because role assignment includes role inheritance, it should be expressed as an extended attribute. Although negative assignment (i.e. *role*! = *R*) can be expressed by reusing the NOT operator, there is no suitable extended leaf node that can express positive assignment (i.e. *role* = *R*). The second problem involves how to express a DC-RBAC access policy (as described in Section 4.2) using the extended tree of ECP-ABE. This is necessary because DC-RBAC and ECP-ABE have different policy models. To solve these problems, we first define a new threshold value for the operator node in ECP-ABE so it can support role assignment. Then, we present a policy mapping model to transform a DC-RBAC policy into an equivalent extended tree form.



DATA-CENTRIC RBAC MODEL

The RBAC model simplifies the management of user permissions in a system. However, as mentioned in Section 1, in the context of self-contained data protection, the RBAC model needs to be able to describe fine-grained access policies that are appropriate to specific data and support arbitrary constraints. In other words, data owners should not only be able to specify access policies for data objects at the role-level but also define other necessary constraints. To meet these requirements, a data-centric RBAC (DCRBAC) model is needed. The DC-RBAC model should support role assignments, inheritance and constraints. It may appear that DC-RBAC is quite similar to RBAC3 which is a consolidation of RBAC1 and RBAC2. However, constraints in DC-RBAC and RBAC3 are quite different. The constraints in RBAC3 roughly include 4 cases: (1) mutually exclusive roles (i.e. separation of duties); (2) cardinality constraints (i.e. limiting the number of users assigned to a role and the number of roles assigned to a permission); (3) prerequisite constraints (i.e., a user can be assigned to a role *A* only if that user is already assigned to role *B*, and permission *p* can be assigned to a role *A* only if role *A* already possesses permission *q*); and (4) constraints associated with sessions, such as the number of sessions that a user can have active at the same time. Clearly, RBAC3 defines its policies at the system level to manage user's privileges for multiple data objects. Its goal is to protect the security of the whole system. In DC-RBAC, the situation is different—the security objective of the system is achieved by protecting each data object. Therefore, the security requirement of each data object becomes the basis of a DC-RBAC policy. Because RBAC3 and DC-RBAC focus on different goals, the constraint compositions (which are important parts in a policy) are quite dissimilar. Regarding the 4 types of constraints in RBAC3, the first constraint can be expressed using the NOT operator in DC-RBAC; the parts of the second and third constraints associated with role assignment should be kept in DC-RBAC, while the parts associated with permission assignment will be abandoned; and the fourth constraint is also abandoned since sessions are no longer needed in DCRBAC.

Structure of DC-RBAC

The DC-RBAC model consists of five sets of entities called *data* (*D*), *users* (*U*), *roles* (*R*), *user attribute constraints* (*Ac*) and *environment constraints* (*Ec*), as shown in Fig. 2. *Data* represents a data object that needs to be protected. *users* are human beings who want to access the protected data. *roles, user attribute constraints* and *environment constraints* together constitute the access policy of the *data*.

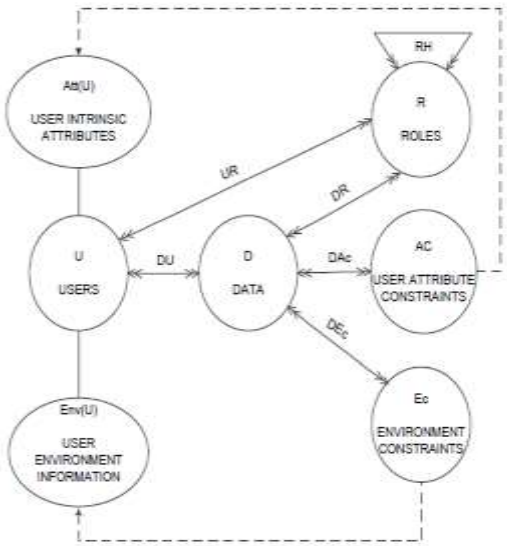
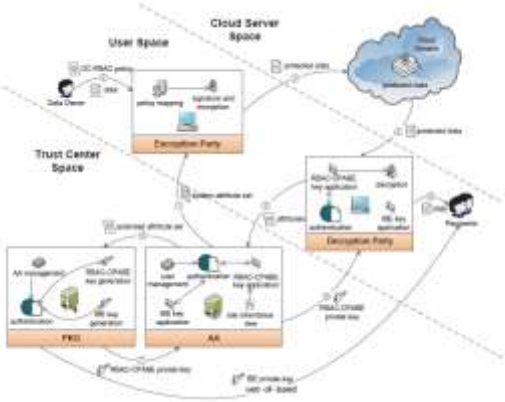


Fig. 2. *Data* represents a data object that needs to be protected



Implemented framework of RBAC-CPABE

Encryption Party. Data owners define access policies and encrypt data in the Encryption Party. To publish data to a cloud server, the data owner uses the data and the DC-RBAC access policy as input. Then, the access policy is mapped to the equivalent extended tree with the *policy mapping module*. Next, the data is signed with the user's IBE private key and hybrid encryption is enforced using the *signature and encryption module*. More specifically, the data is encrypted with AES while the private key of AES is encrypted by RBAC-CPABE using the access policy tree. Finally, the ciphertext, consisting of the AES ciphertext, the RBAC-CPABE ciphertext, the access tree and the signature, is published to the cloud server.

Decryption Party. Data access is achieved through the Decryption Party. The data access process consists of two integral steps as described in Section 5.3.2 (i.e. private key application and data decryption). Using the *RBAC-CPABE private key application module*, the leaf nodes and extended leaf nodes of the access tree attached in the ciphertext are extracted and sent to AA along with the user's identity, forming a request to apply for an RBAC-CPABE private key. Before sending, the message is signed with the user's IBE private key.

Users without an IBE private key must first apply for one through the *IBE private key application module*. After receiving the message from AA, the Decryption Party verifies the signature with the *authentication module* and then extracts the RBAC-CPABE private key. If the user's attributes satisfy the access policy, the *decryption module* will be able to decrypt the RBAC-CPABE ciphertext to obtain the AES private key with which the original data can be decrypted.

AA. The AA is responsible for authenticating users' attributes and invoking PKG to generate private keys. When receiving a message from a user, AA first verifies whether the message is from a valid user using the a*uthentication module*. If it is a valid message, AA analyzes the request type which can be either an IBE private key request or a RBACCPABE private key request. If the request is for an IBE private key, AA extracts the identity of the user and sends it to PKG through the *IBE private key application module*. If the request is for a RBAC-CPABE private key, AA extracts the user's information through the *management module* with the user's identity.

PKG. The main function of PKG in our framework is to generate private keys. Similar to AA, after receiving a message, PKG first verifies whether the message is from a valid AA using the *authentication module* and *AA management module*. When the request is valid, PKG generates the private key using either the *IBE private key generation module* and the *RBAC-CPABE private key generation module* according to the request type. The IBE private key is distributed physically, while the RBAC-CPABE private key is returned to the AA after being signed.

CONCLUSIONS

To address the data protection problem in cloud computing, we propose and implement a role-based self-contained data protection scheme called RBAC-CPABE. Based on the classic RBAC model, we first propose a data-centric access control model, DC-RBAC, which allows the data owner to specify individualized RBAC policies for each data object. Besides role-level constraints, DC-RBAC also contains user attribute constraints and environment constraints, which correspond to information about the authorized users and contextual information about the environment, respectively. Hence, DC-RBAC achieves more flexible and fine-grained access control. Next, to construct the self-contained data protection mechanism, we fuse the DC-RBAC into ECP-ABE by extending ECP-ABE and defining a policy mapping model. By using RBAC-CPABE, information contained in the data itself determines whether users are authorized to perform decryption instead of relying on other parties. Besides ECPABE, RBAC-CPABE also can be constructed based on other tree-based ABE scheme to achieve the specific functionality of the ABE scheme. A security analysis and experiment results indicate that RBAC-CPABE does not add any security risk or computational overhead compared to the CP-ABE scheme on which it is based, but it substantially improves the access control capability. Hence, RBAC-CPABE can be used in clouds to achieve efficient protection for outsourced data.

REFERENCES

[1] C. S. Alliance. (2011) Security guidance for critical areas of focus in cloud computing v3.0. [Online]. Available: https://downloads. cloudsecurityalliance.org/initiatives/guidance/csaguide.v3.0.pdf

[2] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology–CRYPTO*. California, USA: Springer Berlin Heidelberg, 19-23 August 2001, pp. 213–229.

[3] Y. Zhu, G.-J. Ahn, H. Hu, and H.Wang, "Cryptographic role-based security mechanisms based on role-key hierarchy," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*. Beijing, China: ACM, 13-16 April 2010, pp. 314–319.

[4] Y. Zhu, H.-X. Hu, G.-J. Ahn, H.-X. Wang, and S.-B. Wang, "Provably secure role-based encryption with revocation mechanism," *Journal of Computer Science and Technology*, vol. 26, no. 4, pp. 697– 710, 2011.

[5] Y. Zhu, G. J. Ahn, H. Hu, D. Ma, and S. Wang, "Role-based cryptosystem: A new cryptographic rbac system based on role-key hierarchy," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 12, pp. 2138–2153, 2013.

[6] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology–EUROCRYPT 2005*, vol. 3494. Aarhus, Denmark: Springer Berlin Heidelberg, 22-26 May 2005, pp. 457– 473.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*. Alexandria, Virginia, USA: ACM, 30 October-3 November 2006, pp. 89–98.

[8] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy*. Berkeley, CA: IEEE, 20-23 May 2007, pp. 321–334.

[9] Y. Zhu, D. Huang, C. J. Hu, and X.Wang, "From rbac to abac: Constructing
flexible data access control for cloud storage services," *IEEE Transactions on Services Computing*, vol. 8, no. 4, pp. 601–616, July 2015.

[10] B. Lang, R. Xu, and Y. Duan, "Extending the ciphertext-policy attribute based encryption scheme for supporting flexible access control," in *Proceedings of the 10th International Conference on Security and Cryptography*. Reykjavik, Iceland: IEEE, 29-31 July 2013, pp. 1–11.

[11] "Self-contained data protection scheme based on cp-abe," *E-Business and Telecommunications*, vol. 456, pp. 306–321, 2014.

[12] D. Ferraiolo and R. Kuhn, "Role-based access control," in *15th National Computer Security Conference*. Baltimore, Maryland: National Institute of Standards and Technology, 13-16 October 1992, p. 554´lC563.

[13] J. Crampton, "Cryptographic enforcement of role-based access control," in *Formal Aspects of Security and Trust*. Pisa, Italy: Springer Berlin Heidelberg, September 16-17 2011, pp. 191–205.

[14] L. Zhou, V. Varadharajan, and M. Hitchens, "Enforcing role-based access control for secure data storage in the cloud," *The Computer Journal*, vol. 54, no. 10, pp. 1675–1687, 2011.

[15] C. Hong, Z. Lv, M. Zhang, and D. Feng, "A secure and efficient role-based access policy towards cryptographic cloud storage," in *12th International Conference on Web–Age Information Management*, vol. 6897. Wuhan, China: Springer Berlin Heidelberg, 14-16 September 2011, pp. 264–276.