Recent Advances in Technology and Engineering (RATE-2017)
6th National Conference by TJIT, Bangalore

International Journal of Science, Engineering and Technology
An Open Access Journal

SecRBAC: Secure data in the Clouds

[1]Chaitra R, [2]Menaka A

Abstract

Most current security solutions are based on perimeter security. However, Cloud computing breaks the organization perimeters. When data resides in the Cloud, they reside outside the organizational bounds. This leads users to a loos of control over their data and raises reasonable security concerns that slow down the adoption of Cloud computing. Is the Cloud service provider accessing the data? Is it legitimately applying the access control policy defined by the user? This paper presents a data-centric access control solution with enriched role-based expressiveness in which security is focused on protecting user data regardless the Cloud service provider that holds it. Novel identity-based and proxy re-encryption techniques are used to protect the authorization model. Data is encrypted and authorization rules are cryptographically protected to preserve user data against the service provider access or misbehavior. The authorization model provides high expressiveness with role hierarchy and resource hierarchy support. The solution takes advantage of the logic formalism provided by Semantic Web technologies, which enables advanced rule management like semantic conflict detection. A proof of concept implementation has been developed and a working prototypical deployment of the proposal has been integrated within Google services.

Keywords : cloud security,data ownership, encryption

Introduction

SECURITY is one of the main user concerns for the adoption of Cloud computing. Moving data to the Cloud usually implies relying on the Cloud Service Provider (CSP) for data protection. Although this is usually managed based on legal or Service Level Agreements (SLA), the CSP could potentially access the data or even provide it to third parties. Moreover, one should trust the CSP to legitimately apply the access control rules defined by the data owner for other users. The problem becomes even more complex in Intercloud scenarios where data may flow from one CSP to another. Users may loss control on their data. Even the trust on the federated CSPs is outside the control of the data owner. This situation leads to rethink about data security approaches and to move to a data-centric approach where data are self-protected whenever they reside. Encryption is the most widely used method to protect data in the Cloud. In fact, the Cloud Security Alliance security guidance recommends data to be protected at rest, in motion and in use [1]. Encrypting data avoids undesired accesses. However, it entails new issues related to access control management. A rule-based approach would be desirable to provide expressiveness. However, this supposes a big challenge for a data-centric approach since data has no computation capabilities by itself. It is not able to enforce or compute any access control rule or policy. This raises the issue of policy decision for a self-protected data package: who should evaluate the rules upon an access request? The first choice would be to have them evaluated by the CSP, but it could potentially bypass the rules. Another option would be to have rules evaluated by the data owner, but this implies that either data could not be shared or the owner should be online to take a decision for each access request. To overcome the aforementioned issues, several proposals [2] [3] [4] try to provide data-centric solutions based on novel cryptographic mechanisms applying Attribute based Encryption (ABE) [5]. These solutions are based on Attribute-based Access Control (ABAC), in which privileges are granted to users according to a set of attributes. There is a long-standing debate in the IT community about whether Role-based Access Control (RBAC) [6] or ABAC is a better model for authorization [7] [8] [9].Without entering into this debate, both approaches have their own pros and cons. To the best of our knowledge, there is no data-centric approach providing an RBAC model for access control in which data is encrypted and self-protected. The proposal in this paper supposes a first solution for a data-centric RBAC approach, offering an alternative to the ABAC model. An RBAC approach would be closer to current access control methods, resulting more natural to apply for access control enforcement than ABE-based mechanisms. In terms of expressiveness, it is said that ABAC supersedes RBAC since roles can be represented as attributes. However, when it comes to data-centric approaches in which data is encrypted, ABAC solutions are constrained by the expressiveness of ABE schemes. The cryptographic operations used in ABE usually restrict the level of expressiveness for access control rules. For instance, role hierarchy and object hierarchy capabilities cannot be achieved by current ABE schemes. Moreover, they usually lack some combination ith a user-centric approach for the access control policy, where common authorization-related elements like definition of users or role assignments could be shared by different pieces of data from the same data owner. This paper presents SecRBAC, a data-centric access control solution for self-protected data that can run in untrusted CSPs and provides extended Role-Based Access Control expressiveness. The proposed authorization solution provides a rule-based approach following the RBAC scheme, where roles are used to ease the management of access to the resources. This approach can help to control and manage security and to deal with the complexity of managing access control in Cloud computing. The authorization model supports role and resource hierarchies, providing more expressiveness to the rules by enabling the definition of simple but powerful rules that apply to several users and resources thanks to privilege propagation through roles and hierarchies.

Existing System

Encryption is the most widely used method to protect data in the Cloud. In fact, the Cloud Security Alliance security guidance recommends data to be protected at rest, in motion and in use. Encrypting data avoids undesired accesses. However, it entails new issues related to access control management. A rule-based approach would be desirable to provide expressiveness. But this supposes a big challenge for a data-centric approach since data has no computation capabilities by itself. It is not able to enforce or compute any access control rule or policy.

Proposed System

This paper presents SecRBAC, a data-centric access control solution for self-protected data that can run in untrusted CSPs and provides extended Role-Based Access Control expressiveness. The proposed authorization solution provides a rule-based approach following the RBAC scheme, where roles are used to ease the management of access to the resources. This approach can help to control and manage security and to deal with the complexity of managing access control in Cloud computing. Role and resource hierarchies are supported by the authorization model, providing more expressiveness to the rules by enabling the definition of simple but powerful rules that apply to several users.

And resources thanks to privilege propagation through roles and hierarchies.

Related Work

Different approaches can be found in the literature to retain control over authorization in Cloud computing. In [13] authors propose to keep the authorization decisions taken by the data owner. The access model is not published to the Cloud but kept secure on the data owner premises. However, in this approach, the CSP becomes a mere storage system and the data owner should be online to process access requests from users. Another approach from [14] deals with this issue by enabling a plug-in mechanism in the CSP that allows data owners to deploy their own security modules.

This permits to control the authorization mechanisms used within a CSP. However, it does not establish how the authorization model should be protected, so the CSP could potentially infer information and access the data. Moreover, this approach does not cover Inter-cloud scenarios, since the plug-in module should be deployed to different CSPs. Additionally, these approaches do not protect data with encryption methods. In the proposed SecRBAC solution, data encryption is used to prevent the CSP to access the data or to release it bypassing the authorization mechanism. However, applying data encryption implies additional challenges related to authorization expressiveness. Following a straightforward approach, one can include data in a package encrypted for the intended users. This is usually done when sending a file or document to a specific receiver and ensures that only the receiver with the appropriate key is able to decrypt it. From an authorization point of view, this can be seen as a simple rule where only the user with privilege to access the data will be able to decrypt it (i.e. the one owning the key). However, no access control expressiveness is provided by this approach. Only that simple rule can be enforced and just one single rule can apply to each data package. Thus, multiple encrypted copies should be created in order to deliver the same data to different receivers. To cope with these issues, SecRBAC follows a data-centric approach that is able to cryptographically protect the data while providing access control capabilities.

Several data-centric approaches, mostly based on Attribute- based Encryption (ABE) [5], have arisen for data protection in the Cloud [4]. In ABE, the encrypted cipher text is labeled with a set of attributes by the data owner. Users also have a set of attributes defined in their private keys. They would be able to access data (i.e. decrypt it) or not depending on the match between cipher text and key attributes. The set of attributes needed by a user to decrypt the data is defined by an access structure, which is specified as a tree with AND and OR nodes. There are two main approaches for ABE depending on where the access structure resides: Key-Policy ABE (KP-ABE) [5] and Cipher text-Policy ABE (CP-ABE) [3]. In KP-ABE, the access structure or policy is defined within the private keys of users. This allows encrypting data labeled with attributes and then controlling the access to such data by delivering the appropriate keys to users. However, in this case the policy is really defined by the key issuer instead of the encryptor of data, i.e. the data owner. Therefore, the data owner should trust the key issuer for this to properly generate an adequate access policy. To solve this issue, CP-ABE proposes to include the access structure within the ciperthext, which is under control of the data owner. Then, the key issuer just asserts the attributes of users by including them in private keys. However, either in KP-ABE or CP-ABE, the expressiveness of the access control policy is limited to combinations of AND-ed or OR-ed attributes. The data-centric solution presented in this paper goes a step forward in terms of expressiveness, providing a rule-based approach following the RBAC scheme that is not tied to the limitations of current ABE approaches.

Different proposals have been also developed to try to alleviate ABE expressiveness limitations. Authors in [15] propose a solution based on CP-ABE with support for sets of attributes called Ciphertext Policy Attribute Set Based Encryption (CP-ASBE). Attributes are organized in a recursive set structure and access policies can be defined upon a single set or combining attributes from multiple sets. This enables the definition of compound attributes and specification of policies that affect the attributes of a set. An approach named Hierarchical Attribute-based Encryption is presented in [16]. It uses a hierarchical generation of keys to achieve fine-grain access control, scalability and delegation.

However, this approach implies that the same root domain authority should manage attributes. In [17], authors extend CP-ASBE with a hierarchical structure to users in order to improve scalability and flexibility. This approach provides a hierarchical solution for users within a domain, which is achieved by a hierarchical key structure. Another approach is Flexible and Efficient Access Control Scheme (FEACS) [2]. It is based on KP-ABE and provides an access control structure represented by a formula involving AND, OR and NOT, enabling more expressiveness for KP-ABE.

Data-Centric Solution For Data Protection In The Cloud

In the protected authorization model specified in Section 5.1, it should be observed that data is not encrypted with the data owner identity, but with the object's own identity (e.g. ido1). This follows a data-centric approach for data protection, in which data is encrypted with its own key under the cryptographic scheme. If a pure PRE scheme were used, the object would be also encrypted using its own key pair. On another hand, a user-centric approach is used for the authorization rules, where the data owner for its data defines a unified access control policy. This allows to share common definitions and to greatly simplify access control management, getting the most from role hierarchy and resource hierarchy capabilities. An architecture is also proposed for the deployment within CSPs. This architecture takes into consideration the different elements that should be deployed in order to give an overview of how access to protected data is done in this approach. Fig. 2 depicts the proposed architecture.

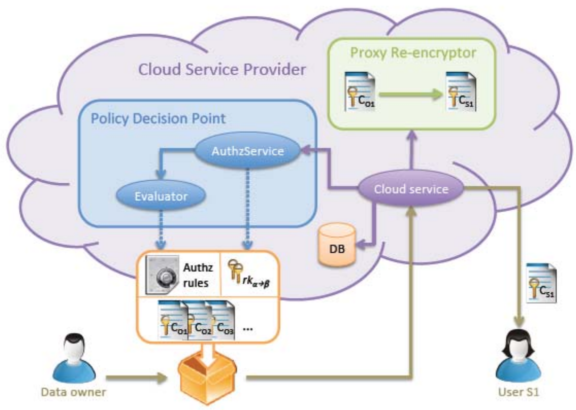

Fig. 2. Architecture for deployment in a CSP

When the data owner generates moving data to the cloud, a self-protected package. This package contains the encrypted data objects, the authorization rules and the corresponding re-encryption keys.

Data objects are encrypted before uploading them to the Cloud in order to prevent the CSP to access them. This is done by data owners by using the encrypt () function (3). According to Def. 8, data should be encrypted using the identity ido1 of the object being uploaded o1. A digital envelope approach can be applied to protect data objects instead of direct encryption. This would enhance cryptographic operations like re-encryptions for large data objects.

This approach consists in using a symmetric encryption algorithm (e.g. AES) to protect the data object itself. The encryption of data is done with a random symmetric key generated for the purpose of a single encryption. Then, this key is encrypted with the encrypt () function. With this procedure, potentially big objects (e.g. large documents) are encrypted using symmetric cryptography, whose algorithms are more efficient. In turn, more costly operations are only applied to the keys used for the symmetric encryption, which are usually small pieces of data of some bytes length.

Authorization rules are defined by the data owner and directly mapped into the authorization model. This is done by including the corresponding elements in the binary relations. For instance, a rule that grants a subject s1 access to an object o1 would add an element g1 = (s1; o1) to the binary relation Ga. Depending on the representation of the authorization model, the corresponding rule should be generated. Following the example, in case of an ontology, a granted object property linking s1 to o1 should be included.

Re-encryption keys are generated by the rkgen() function (4) for every authorization rule. The data owner using the Master Secret Key and the identities of the involved authorization elements can only generate them.

Note that, although the rkgen() function initially takes the Secret Key of one element, this key can be really computed on demand using the keygen() function, which only needs the MSK and the identity. That is, by (2) and (4) rkgen (p; keygen (p; msk; id_) ; id_; id_) ! rk_!_. This is the main advantage of using IBE instead of a pure PRE scheme. The usage of IBE enables the generation of reencryption keys with just the MSK and the identities of the authorization elements, avoiding the need to store dedicated public and private key pairs for each authorization element.

The re-encryption keys can be considered as cryptographic tokens provided by data owners in order to enable to perform operations over the authorization model. That is, a CSP will not be able to effectively apply any rule that has not been legitimately defined by the data owner. If a CSP tries to do so, it would result in a re-encryption chain that would miss some re-encryption keys of the path, being that re-encryption chain useless to access the data.

Applying this data-centric approach results in an self contained protected object, which can be released to the Cloud and only authorized users could access the data object. This can be useful for Inter-cloud scenarios, where data can travel through different CSPs. Before generating the first package, the data owner should initialize the cryptographic suite by using the setup() function (1). As result, the mask is obtained and should be kept private. In turn, the parameters p returned by the setup() function should be made available for other users to execute cryptographic operations. This could be simply done by publishing them on the CSP. However, a better approach would be to include this information together with the encrypted data. Standard cryptographic formats usually have fields defined to this purpose. For instance, the Cryptographic Message Syntax (CMS) [22] allows to include an initialization vector which contains information to be used to initialize the cryptographic suite.

Regarding the CSP, there are two main modules to be deployed: a Policy Decision Point (PDP) which manages the authorization model and a Proxy Re-encryptor that performs the cryptographic operations. Cloud services offered by the CSP interact with these two modules in order to securely manage the data. They make use of a database to store the protected packages uploaded by data owners. Thus, it contains the information of these packages, i.e. encrypted data objects, authorization rules and re-encryption keys. It can also contain the parameters p to initialize the cryptographic functions. The information can be kept in data packages as provided by the data owner or it could be stored on any other format that facilitates data processing to the CSP. An authorization service (Authz Service) acts as entry point to the PDP for Cloud services allowing querying it for authorization decisions. This module takes decisions upon a request from a user s1 to access to a piece of data o1 managed by the service. These decisions usually return an access granted or denied statement. For granted accesses, the response also contains the re-encryption chain that should be applied, together with the needed re-encryption keys. This information allows to re-encrypt from co1 as provided by the data owner to cs1 , which can be decrypted by the requesting user. The service passes this information together with co1 to the Proxy Re-encryptor for this to perform the re-encryption operations. It results in cs1 , which is sent to the requesting user. Making use of its own secret key sks1 the user can decrypt the data with the decrypt() function (6). Note that during this process, the CSP is not able to access the data since it only applies a set of re-encrypt() functions which do not disclose any information about the data being processed. The PDP considers two different sources of information: authorization rules and re-encryption keys. This information is provided in the protected packages uploaded by the data owner. As mentioned before, the former contains a representation of the protected authorization model as described in section Section 5.1. This can be the ontology explained in Section 5.2 or any other valid representation of the model. The latter contains the set K of re-encryption keys. The AuthzService makes use of an evaluator module in charge of making the computations over the authorization model. This module depends on the representation of the model that is being used. In case of an OWL ontology, it corresponds to a reasoner engine able to perform queries over the model. The results of this module are processed by the AuthzService to form an authorization decision. In case of a positive decision, it also obtains the corresponding authorization chain and the re-encryption keys for such a chain. This information conforms the output of the AuthzService for a request that is passed to the Proxy Re-encryptor module.

Conclusion

A data-centric authorization solution has been proposed for the secure protection of data in the Cloud. SecRBAC allows managing authorization following a rule-based approach and provides enriched role-based expressiveness including role and object hierarchies. Access control computations are delegated to the CSP, being this not only unable to access the data, but also unable to release it to unauthorized parties. Advanced cryptographic techniques have been applied to protect the authorization model. A re-encryption key complement each authorization rule as cryptographic token to protect data against CSP misbehavior. The solution is independent of any PRE scheme or implementation as far as three specific features are supported. A concrete IBPRE scheme has been used in this paper in order to provide a comprehensive and feasible solution. A proposal based on Semantic Web technologies has been exposed for the representation and evaluation of the authorization model. It makes use of the semantic features of on tologies and the computational capabilities of reasoners to specify and evaluate the model. This also enables the application of advanced techniques such as conflict detection and resolution methods. Guidelines for deployment in a Cloud Service Provider have been also given, including an hybrid approach compatible with Public Key Cryptography that enables the usage of standard PKI for key management and distribution. A prototypical implementation of the proposal has been also developed and exposed in this paper, together with some experimental results. Future lines of research include the analysis of novel cryptographic techniques that could enable the secure modification and deletion of data in the Cloud. This would allow extending the privileges of the authorization model with more actions like modify and delete. Another interesting point is the obfuscation of the authorization model for privacy reasons. Although the usage of pseudonyms is proposed advanced.

References

[1] “SPARQL 1.1 overview,” World Wide Web Consortium (W3C), W3C Recommendation, Mar. 2013.

[2] R. Housley, “Cryptographic message syntax (CMS),” Internet Engineering Task Force (IETF), RFC 5652, Sep. 2009.

[3] E.-J. G. Dan Boneh and T. Matsuo, “Proposal for p1363.3 Proxy Re-encryption,” Aug. 2006.

[4] O. K. J. Mohammad, S. Abbas, E. M. El-Horbaty, and A. M. Salem, “Innovative method for enhancing key generation and management inRefer simply to the reference number, as in [5]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Author’s details

Ms. Chaitra .R received BCA from Jain University, 2015and pursuing MCA (master of computer applications) with specialization in Storage and Cloud Technology from Jain university, she is VMware certified .Her main area of interest in cloud computing.

Ms. A. Menaka received B.Sc Industrial Electronics from Bharathidasan University, 2002, MCA (Master of Computer Applications from Avinashilingam Deemed University, 2005, and M.Phil Computer Science in Bharathiar University, 2014. She is MCP, MCSE, MCITP and MCT certified and has 5 years of experience in training field towards Microsoft certifications. She has 6+ years of teaching experience in academics and published varies research papers in field of networking. Her main area of interest is Networking, Cloud computing and infrastructure management.