Semantically Rich, Oblivious Access Control Using ABAC for Secure Cloud Storage

Maithilee Joshi, Sudip Mittal, Karuna P. Joshi and Tim Finin
University of Maryland, Baltimore County, Baltimore, MD 21250, USA
Email: {maithi1,smittal1,kjoshi1,finin}@umbc.edu

***Abstract*—Securing their critical documents on the cloud from data threats is a major challenge faced by organizations today. Controlling and limiting access to such documents requires a robust and trustworthy access control mechanism. In this paper, we propose a semantically rich access control system that employs an access broker module to evaluate access decisions based on rules generated using the organizations confidentiality policies. The proposed system analyzes the multi-valued attributes of the user making the request and the requested document that is stored on a cloud service platform, before making an access decision. Furthermore, our system guarantees an end-to-end oblivious data transaction between the organization and the cloud service provider using oblivious storage techniques. Thus, an organization can use our system to secure their documents as well as obscure their access pattern details from an untrusted cloud service provider.**

***Index Terms*—Access Control, Access Broker, Ontologies, Confidentiality Policy, Oblivious Storage, Cloud Computing**

I. Introduction

Organizations are rapidly adopting cloud based services, like Box[1], GSuite[2], Dropbox[3], etc. to better manage and collaborate on their official documents. These documents need to be kept safe from any external or internal threats. For instance, health-care providers maintain personal health records of their patients which must be kept private and secure and at the same time should be available to authorized users, like patients, who will access them from outside organizational boundaries. To ensure the security and privacy of document(s), organizations define their own document confidentiality policies. These policies state the rules and regulations that must be enforced and abided by the employees to protect the intellectual property of the organization. Document confidentiality policies include access control mechanisms that determine: lists of authorized groups; users and roles; encryption policies; security classification for documents; and folders used to manage these documents. To ensure high confidentiality, organizations want no unauthorized user to be able to access their documents. This will require not only that the document's contents are encrypted, but that access levels, security classification and usage patterns are also obfuscated from unauthorized users on the cloud, including the cloud service provider.

Access control mechanisms are developed to limit the correct access to an authorized person. Some naive systems maintain a fixed access control list (ACL) on each document based on which access control is selected [1]. In more advanced systems, access control is implemented using the concepts of Role Based Access Control (RBAC) [2]. However, for securing complex critical documents and organizational structures these naive or RBAC systems do not suffice. Attribute Based Access Control (ABAC) is an improvement over RBAC. ABAC uses a number of varied and multi-valued attributes to evaluate an access control decision [3].

Most cloud providers currently provide encryption mechanisms to store cloud data. However, the cloud provider can still view the usage pattern of a document. Also, the cloud provider's Identity and Access Management (IAM) module gives full knowledge of the user profiles and their access levels to the documents stored on the cloud instance [4].

Cloud consumers, while storing their data on the cloud, usually regard cloud providers under two main adversary models [5] - the Honest But Curious (HBC) adversary model where providers run the programs and algorithms correctly, but might look at the information passed between entities; and the Malicious adversary model where providers behave in an arbitrary manner that may be hostile to the cloud customer. In this paper, we consider the scenario where the cloud storage provider is not completely trusted by the consumer, that is the malicious adversary model. In this use case, the cloud consumers want their cloud data as well as it's usage pattern and access levels to be oblivious to the provider.

To do so, we present a solution that enforces the principles of Edge Computing [6]. Edge computing refers to the notion of performing data operations before moving it to the cloud. Accordingly, in the solution we present, we enforce an access control mechanism on the data inside the organizational boundary which we define to be the "edge" in our system. This makes sure that users are authenticated inside the organizational boundary, in turn preserving the obliviousness of their identity. Along with this, we have implemented a robust encryption mechanism inside the organizational boundary that ensures data protection against any kind of data threats before moving it to the cloud. Thus this organizational boundary proves to be a strong "edge" for securing data privacy.

In this paper, we describe our system which allows a semantically rich access control design for managing documents on the cloud. For our use case, we consider an organization's employee(s) requesting access to critical organizational document(s) that is(are) stored on a cloud service platform. We

[1] www.box.com
[2] gsuite.google.com/together/
[3] www.dropbox.com

have created a complex ontology to describe critical attributes of documents and users. This ontology forms the underlying framework of the "Access Broker" module that manages the document access. The organizational confidentiality policies are represented in our system using Semantic Web Rule Language (SWRL) rules [4]. To protect third party and even cloud provider intrusion, the proposed system guarantees a completely oblivious end-to-end data exchange between the organization and the cloud service provider. Using encryption techniques like Oblivious RAM (ORAM) [7] [8], this system completely obscures the identity of the employee accessing the document, the contents of the document as well as the data access patterns from any third party or the cloud service provider.

In our system, a user requests a document which is assumed to be with the cloud service provider. In the first step, a user is assessed by the Access Broker module on various user and document attributes by employing a number of semantic web rules which are based on the confidentiality policy defined by the organization. On the Access Broker providing an affirmative decision on the access control, the further modules of the system engage in extracting the document from the cloud service provider. For security and privacy purposes we used the ORAM encryption mechanism which obscures the data, the data access patterns as well as the user identities from the cloud service provider or any other intruder.

The primary beneficiaries of our system will be various organizations who create their own confidentiality policies for controlling the access to their critical documents stored on the cloud platform. This is because the system evaluates an access decision not only on a single parameter for both documents and users in the organization, but also on several attributes like the department, work location, clearance level, etc. Thus, organizations can secure their critical documents by allowing access to restricted employees only, and in turn reduce the risk of an internal attack.

The rest of the paper is organized as follows – we discuss related work in Section II and our system architecture in Section III. We evaluate and present our conclusions in Sections IV & V.

II. Related Work

Various approaches like Oblivious RAM (ORAM) [9], [10] and Oblivious Storage [11] have been proposed to address obliviousness of data. The seminal work on the topic is Goldreich and Ostrovsky [9], and since then, much work (e.g., [10], [12]) has focused on improving the efficiency of ORAM in terms of space, time, and communication cost. Moreover, considerable research work is conducted on storing data where cloud providers are untrusted [13], [7].

Narkhede et al. from our research group have developed a semantically rich application that enables cloud users to store their cloud data in an oblivious fashion. The ObliviCloudManager [8] is a web application that allows users to define and enforce policies on the documents to be stored on the cloud in an oblivious fashion. The system used an oblivious map data structure (ODS map) [14] as the back-end to store files. Its primary features are a simple key-value architecture and strong privacy guarantee. As a key-value store, it represents a set of (key, value) items and supports inserting an item, searching for an item containing a given key, and deleting an item with a given key. In our context, a data item corresponds to a file, with the file-name as the keyword and the file contents as the value. The map structure provides strong privacy guarantee of obliviousness. The network traffic to a server can reveal which raw blocks are being read and written to an attacker or the server itself. This access pattern may leak sensitive information about the underlying stored data, such as keyword search queries or encryption keys. However, the ObliviCloudManager system has not implemented any access control mechanisms apart from full access and needs additional research that we are proposing in our design. We have also been looking at ways to design access control mechanisms using semantically rich policy-based approaches.

Nabeel et al. [15] created a system called "CloudMask" which identifies the challenges in securing the Data-as-a-Service (DaaS) model specifically for organizations to enjoy all the benefits of hosting their data in the cloud while at the same time supporting fine-grained and flexible access control for shared data hosted in the cloud. Li et al. [16] presented a way to implement scalable and fine-grained access control systems based on attribute-based encryption (ABE).

Shi et al. [6] have defined the concept of edge computing by highlighting the benefits of performing computations at the data sources site instead of at the cloud provider's end. In this paper, we support the idea of edge computing by performing access control and data encryption operations in the proximity of data i.e. inside the organizational boundary which as previously mentioned is the "edge" in our system.

The commonly known access control models include Discretionary Access Control (DAC) [1], Mandatory Access Control (MAC) [17] and Role Based Access Control (RBAC). However, these models have some shortcomings which limit their practical use. For example, in DAC it is possible that information may be accessed by unauthorized users due to the existence of multiple copies of the information and no control on them. MAC avoids the problem by using the concept of security levels (clearance and classification). The security levels are applicable on each copy of the object. However, few security levels put a restriction on commercial applications. RBAC is based on user roles and associated permissions with each role. RBAC also suffers from problems like role explosion which corresponds to the situation in which different roles may require different permissions. This results in large number of roles and it becomes difficult to manage them. Sharma et al. [3] presented an access control solution using ABAC by representing user properties in an ontology. In this paper, we use a similar concept, however, we have enhanced our solution by encorporating a robust encryption mechanism like ORAM. Apart from that, we consider confidentiality

[4]https://www.w3.org/Submission/SWRL/

policies that are more generic in terms of organizational use.

Attribute Based Access Control (ABAC) model proposed by Jin et al. [18] is a general framework which combines the benefits of DAC, MAC and RBAC and goes beyond their limitations. The model is based on generic attributes which are used to capture identities and access control lists for DAC, clearances and classifications for MAC and roles for RBAC. This model allows for more flexibility for policy specifications as any number of attributes can be added within the same extensible framework. This also solves the shortcomings of the core RBAC model as appropriate attributes such as location, business hours etc. can be added within a unified framework. Dynamic computation of the authorization can be done at the time when the access request is made.

There have been many efforts to model access control policies. XACML [19] which is based on XML specification language, is a general purpose authorization policy model. It enforces access control based on attributes. Rei policy language [20] is another effort which is based on deontic concepts. In Rei, credentials and entity properties like user, agent, etc are associated with access privileges. Another related piece of work is the Rein framework which builds on Rei and also based on N3 rules. CWM reasoning engine is being used for distributed reasoning in Rein. KAoS [21] and ROWLBAC [22] are other works in the related area which are based on OWL. Few research efforts on access control have considered designing models based on semantic web, ABAC, obliviousness [23], [24], [25], [15], [26].

The Web Ontology Language (OWL) [27] provides an efficient way to represent policies formally. It is used to represent security policies [20], [28]. OWL provides an efficient way to write complex ontologies. OWL representation of ABAC_α policies have been presented in [28]. In this work, basic constructs (`User`, `Subject`, `Object`, `Permission`) are defined as OWL classes. OWL properties are used to define `User Attribute`, `Subject Attribute` and `Object Attribute`. The authors have defined models, domains, data and security policies in OWL and used a reasoner to decide what is permitted.

Li et al. [29] in their work on securing the personal health records of patients using fine-grained access control have stressed upon securing the privacy of patient data in a multi-user setting where there are multiple users accessing the data. In this paper, we aim to secure the privacy of documents not restricted to the health records.

III. System Architecture

In this section we describe in detail, our overall system architecture (Figure 1) and various sub-modules. Our system can be divided in two major parts; the first part, is inside the organizational boundary and includes the Access Broker and File Exchange modules. These modules are under organizational control and hence are considered to be 'trusted' entities. The second part, resides outside the organization and includes the 'untrusted' cloud service provider.

Our system has two use cases, when the user wants to "Read" a document and the other one where he wants to "Write" to a document. In the read phase, the user requests access to a document. This request is first assessed by the Access Broker module (Figure 5). The Access Broker gets the query specific instances from the user and document graphs. The access decision is then evaluated by the Access Broker by analyzing ABAC rules defined in accordance to the confidentiality policy. If the Access Broker allows the access request, then the file exchange and encryption module using the ORAM encryption mechanism encrypts the request to the cloud provider. The document is fetched by the file exchange and encryption module and decrypted. The user then "reads" the document.

Similarly, while writing to the document, all modules function as they did while reading. The cloud provider in this case, updates the writes made by the user to the document. As all the data packets are encrypted with ORAM, the data transfer and storage are completely secure and oblivious.

In the next few sections, we will describe each of our system modules in detail.

A. Confidentiality Policy

Confidentiality policy of an organization in an access control setting are specific terms, rules and conditions that are used to determine and evaluate access to proprietary, confidential and valuable information. These policies are generally framed by an authorized regulatory body internal to the organization in question and must be followed by any user interacting with the organization.

Our system allows the organization to declare and exercise their own confidentiality policy. It can generate and implement rules based on multiple confidentiality policies of a given organization. Given a scenario where a user needs to be validated on a number of constraints which belong not just to a single confidentiality policy but multiple ones, in that case, our system appropriately executes the rule by considering both policy rules separately. In another scenario, there could be a possibility of the user and document attribute(s) changing dynamically. In the case when an attribute changes, there could be a chance of a reversal in the access control decision made by the Access Broker. For example, the confidentiality policy for a particular document, apart from other constraints, states that the user requesting the document must have a work location inside the United States. And now this user travels out of the United States. In this case the attribute value of work location of the user has changed and as a result the user will be denied access based upon his/her physical location. We leave the decision for such scenarios on the confidentiality policy itself. If the policy defines rules for such dynamic updates, then the rules generated guarantee provision of a correct dynamic decision.

The Bell LaPadula Model [30], the Biba Model [31] are some of the popularly accepted confidentiality policies. To build a proof of concept, we use the Bell LaPadula as an example confidentiality policy model. The Bell LaPadula model

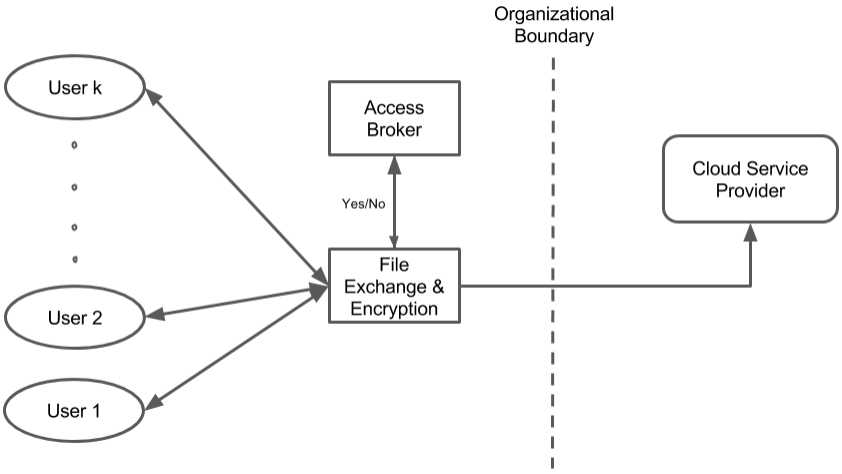

Fig. 1: Overall System Architecture.

characterizes the subject and object into different levels of confidentiality. These levels in decreasing order of importance are (generally, for example) top secret, secret, confidential and unclassified. The model works on the principle of "no read up, no write down" which means that users at any given confidentiality level can write to only those documents that reside at the same confidentiality level or above them, but cannot read any documents residing above their given level.

In our system we used the Bell LaPadula Model [30] to define rules using semantic web techniques (Section III-C). Formally stating, Consider the following sets:

User set,

$$U = \{User_1, User_2, ...User_n\}$$

Document set,

$$F = \{Doc_1, Doc_2, ...Doc_m\}$$

Hierarchy set,

$$H = \{H_1, H_2, H_3, ...H_n\}$$

where,

$$H_1, H_2, ...H_n = \{TS, S, C, U \mid TS > S > C > U\}$$

here,

$$TS = TopSecret,\ S = Secret,$$

$$C = Confidential,\ and\ U = Unclassified.$$

$$\forall User\ U, \exists Hierarchy\ H.$$

Similarly,

$$\forall Document\ D,\ \exists Hierarchy\ H.$$

Now, for every request by a user U and residing at a hierarchy H, the Access Broker applies the predefined rules to decide whether or not to allow the user the requested document access. Rules could be defined as follows,

$$User(User_x, H_y) \wedge Document(Doc_m, H_n) \mid (H_y \leq H_n) \Rightarrow Write\_Only(User_x, Doc_m)$$

$$User(User_x, H_y) \wedge Document(Doc_m, H_n) \mid (H_y \geq H_n) \Rightarrow Read\_Only(User_x, Doc_m)$$

$$User(User_x, H_y) \wedge Document(Doc_m, H_n) \mid (H_y = H_n) \Rightarrow Read\_or\_Write(User_x, Doc_m)$$

In this way, using the semantic approach, the user would either be granted or denied the access to the requested file.

B. Users and Documents

Users are the employees of the organization. Users request a document which they wish to access, change or modify. These requests can be of two types, read and write.

For users, we have built a knowledge graph which describes their attributes inside the organization. These attributes will be used by our system to make access control decisions. Examples of such attributes include the rank of the user inside the organization, his/her clearance level, location of work, department to which he/she belongs, etc. In our system, these attributes are stored as a knowledge graph as shown in Figure 2. An example of a user in the knowledge graph can be seen in Figure 3.

Documents are essentially files that belong to the organization. These documents may contain sensitive information and ensuring authorized and oblivious access to it is the main intention behind our system. All the necessary documents

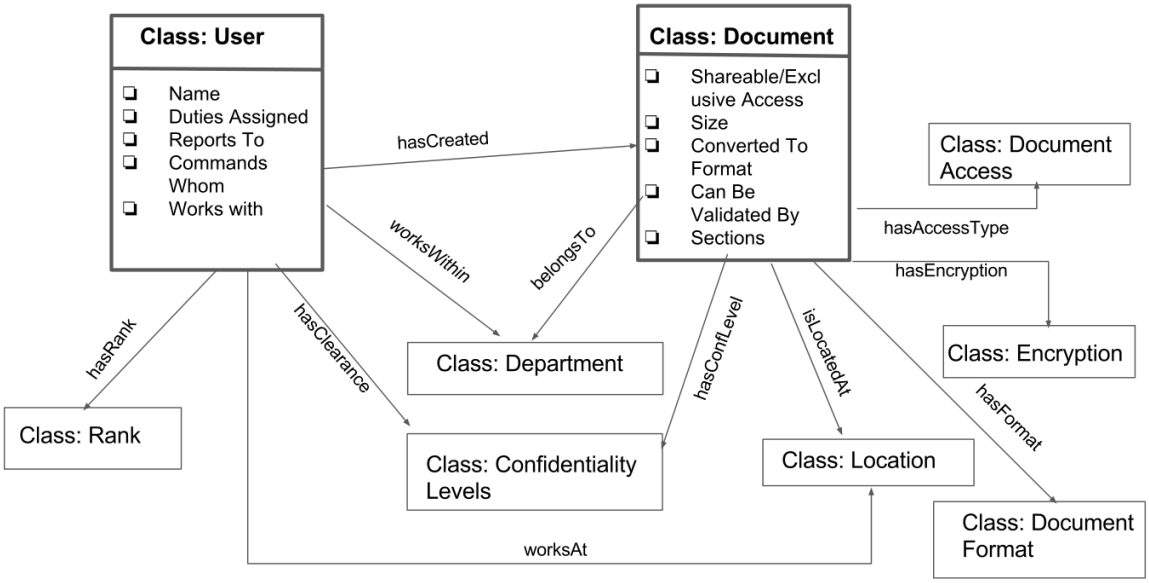

Fig. 2: Ontology Schema.

```
<!--
    http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#Mindy
  -->
<owl:NamedIndividual rdf:about="http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#Mindy">
    <rdf:type rdf:resource="http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#User"/>
    <hasClearance rdf:resource="http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#Secret"/>
    <hasRank rdf:resource="http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#LieutenantCommander"/>
    <worksAt rdf:resource="http://www.semanticweb.org/maithilee/ontologies/2016/10/AccessBroker#PacificFleet"/>
    <hasReadAccess rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</hasReadAccess>
  </owl:NamedIndividual>
```

Fig. 3: Attributes for a user named Mindy, values for each attribute are: hasClearance - Secret, hasRank - LieutenantCommander, worksAt - PacificFleet. Derived property value for hasReadAccess is true.

are assumed to be available from the cloud provider. Each user does not have the same access permissions on all the documents. The confidentiality policy determines the access control mechanism that is, the extent to which any given user has access to a requested document. An example for a document can be seen in Figure 4.

We also built a knowledge graph for documents which describe various attributes of a document like the creation department, confidentiality level, etc. The two primary classes of the ontology are the 'User' class and the 'Document' class. The User class defines all the characteristics and attributes of an employee in an organization. For example, attributes like the work location, duties assigned, team with which the user works, rank he/she holds in the organization, etc. The User class is associated with the Document class via the *hasCreated* property. Like the User class, the Document class too defines all the properties that are associated with any organizational document. For example, the format of the document, size of the document, encryption mechanism used to encrypt it, department to which it belongs, etc.

We have a number of other classes defined which serve as the helper classes. We require these helper classes to reference the complex properties of users and documents respectively. For example, consider the Confidentiality Levels class. This is an associate class that consists of the different levels of confidentiality defined by the confidentiality policy. For example, in the Bell LaPadula model, we have four levels of confidentiality as mentioned in the previous section. Using the "no read up and no write down" policy the Access Broker extracts the value of the confidentiality level associated with the user and document(s) in question from the Confidentiality Levels class and then executes the required rule to give an access control decision. All the properties of users and documents mentioned above reference a different associate class.

Confidentiality Level class includes all the rules that belong to the confidentiality policy being considered by the organization (in our case the Bell LaPadula model). Each user and document entity is categorized into a hierarchical level of confidentiality. A given user is granted access to a document based on the policy of "no read up and no write down" As shown in Figure 2, the User class has an attribute named *hasClearance* associated with the Confidentiality level class. This attribute describes the level at which the user resides with respect to the confidentiality policy obeyed by the organization (in our case

Fig. 4: Attributes for a document named Shipment, values for each attribute are: belongsTo - AtlanticFleet, createdBy - Davis, accessType - Read, hasConfLevel - Secret.

the Bell LaPadula model). On the other hand, the Document class is associated with the Confidentiality Level class via an attribute named *hasConfLevel*. This attribute categorizes a document based on its confidentiality. Policies defined by the Bell LaPadula model decide the access permissions using the values of the attributes *hasClearance* and *hasConfLevel* of the User instance and Document instance, respectively.

C. Access Broker

As shown in Figure 5, this module is responsible for implementing the access control mechanism of the system. The module implements the confidentiality policy in form of SWRL rules. To do this, the Access Broker uses the rules defined on the confidentiality policy. Whenever a user requests access to a file, the Access Broker applies these rules on the user and document knowledge graph instances and makes a binary (yes/no) decision for access control. For example consider the following rule,

```
User(?u) ^ Document(?doc) ^
    ConfidentialityLevels(?c) ^
    ConfidentialityLevels(?l) ^
    hasClearance(?u, ?c) ^ hasValue(?c, ?x) ^
    hasConfLevel(?doc, ?l) ^ hasValue(?l, ?y)
    ^ swrlb:greaterThanOrEqual(?x, ?y)
=>
hasReadAccess(?u, true)
```

This rule evaluates a "read" request by some user on the organization's document. This rule provides an access decision based on the Bell LaPadula model using user and document attributes like the *hasClearance* i.e. the clearance level of the user and *hasConflevel* i.e. the confidentiality level of the document. Suppose we consider some user *Davis* having a *TopSecret* level of clearance. Let the document named *Shipment* have a *Secret* level of confidentiality, then user *Davis* is permitted a read access. On the other hand, if user *Davis* has a clearance level of *Confidential* then the access would have been denied as it would have violated the principles of Bell LaPadula model. This rule evaluates the access control based on only few parameters. This rule does not consider, for example, the rank of the user. In scenarios where rank of the user matters more than the clearance level, a different rule is applied by the Access Broker module. For example there can be a rule like,

```
User(?u) ^ Document(?doc) ^
    ConfidentialityLevels(?c) ^
    ConfidentialityLevels(?l) ^ Rank(?r) ^
    hasClearance(?u, ?c) ^ hasValue(?c, ?x) ^
    hasConfLevel(?doc, ?l) ^ hasValue(?l, ?y)
    ^ hasRank(?u, ?r) ^ hasValue(?r, ?w) ^
    swrlb:greaterThanOrEqual(?x, ?y) ^
    swrlb:lessThanOrEqual(?w, N)
=>
hasReadAccess(?u, true)
```

The above rule allows access only if the user's profile obeys the Bell LaPadula model as well as the rank is above some expected rank. Consider a scenario where there are two users *Davis* and *Mindy*. Suppose *Davis* is a *Captain* of the *Pacific Fleet* having clearance level at *TopSecret*, while *Mindy* is a *Lieutenant Commander* of the *Pacific Fleet* having a clearance level at *Secret*. Let the expected level of rank be that above the rank say, *Captain*. Then according to the above rule, *Mindy* would be granted access to *Shipment* while *Davis* would be denied access to the same document because *Davis's* rank did not match to the ranks above which access is permitted. That is, in-spite of a higher clearance level, user *Davis* was denied access because the document access was permitted to all the officers ranked above *Captain*. In this way, such complex rules assist the Access Broker to evaluate access control decision on each document belonging to the organization.

D. File Exchange and Encryption Module

We have built this module to handle data encryption and communication across the organizational boundary. As shown in Figure 6, the Account and Key Management module handles internal user accounts and the keys required for encrypting and decrypting user requests and data packets. We use the ORAM [7], [8] for encryption. The ORAM encrypts the data in an oblivious manner so that not only is the data oblivious but the data access patterns are also oblivious. The ORAM uses the ODS implementation developed by our collaborators in USNA [14]. ODS uses the map structure that guarantees strong privacy and obliviousness. In our system's write phase, that is

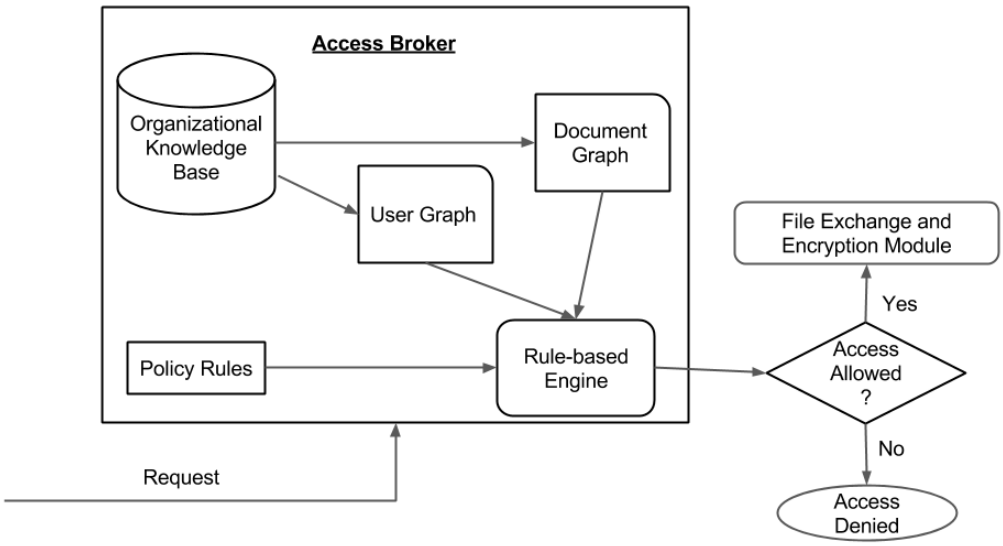

Fig. 5: Access Broker

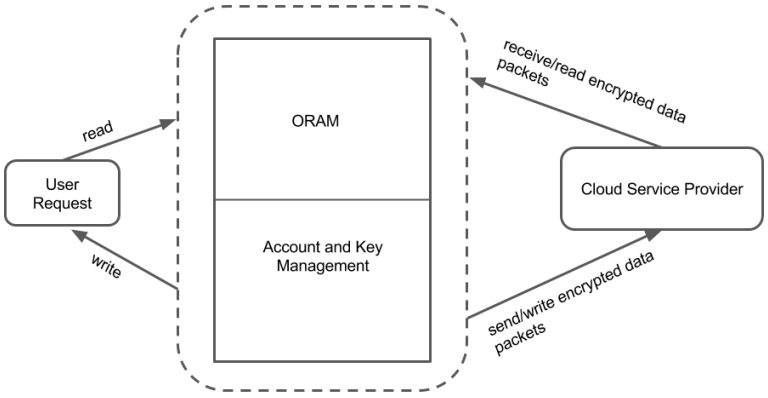

Fig. 6: File Exchange & Encryption Module

when the user requests to change/modify some document of interest, in that case, as shown in Figure 1, the file exchange and encryption module waits for the Access Broker to make a decision on the user‘s request for document access. On receiving a confirmation from the Access Broker, this module initiates a data transfer operation between the user entity and itself. Using ORAM, it then encrypts the received data packets. These encrypted blocks then get transferred to the cloud service provider. In the read phase, the ORAM decrypts the data packets received from the cloud service provider. If the Access Broker permits read access, then the data packets are transferred to the user.

E. Cloud Service Provider

We use the capabilities of a public cloud service provider for hosting the organizational documents. Using our semantic web model of access brokering based on confidentiality policies and the ORAM encryption technique, we ensure that the cloud provider remains oblivious not only to the data it hosts but also to the access patterns of the data. The cloud provider thus has zero knowledge of the identity of the user requesting the document, the data, the access levels of the data or the data access patterns. The cloud provider uses its own mechanism to encrypt, store and replicate the ORAM encrypted data packets. As a result, an extra layer of encryption is applied on the already robust ORAM encryption.

IV. Evaluation

To evaluate our system we conducted a user study where we asked a few domain experts to create scenarios based on the Bell LaPadula confidentiality model. The users were first briefed on the confidentiality policy in detail followed by a question and answer session where they could interact with one of the authors of the paper to address their questions

about the system. After the interaction, the users were tasked to create complex scenarios where they believed the system would fail. The users were tasked with writing these scenarios and then recreated them on the system with some help provided by one of the authors. All the users were able to create their scenarios on the system and were satisfied with the way the system handled their scenarios.

V. Conclusion & Future Work

In this paper we proposed an access control system that would help determine access to an organization's critical documents on a cloud platform. We proposed a semantically rich Access Broker that makes use of the rules based on the organization defined confidentiality policy along with the multiple, variable user and document attributes. The proposed architecture also guarantees obliviousness of user identity, document identity and document access pattern from any external entity by using ORAM based techniques.

In future, we plan to implement an addition to the existing system which would facilitate multiple users of the organization to collaborate on the same document. We plan to refine the Access Broker by including the rules which allow it to assess multiple users and documents based on their attributes to provide an appropriate access decision. Another major direction for us could be implementing access control on different parts of a document.

Acknowledgment

This research was supported by the Office of Naval Research under grants N00014-15-1-2228 and N00014-16-WX-01489. We thank Dr. Seung Geol Choi (USNA), Dr. Adam Aviv (USNA), Dr. Daniel Roche (USNA) and members of the Ebiquity Research Group for their vital feedback.

References

[1] R. S. Sandhu and P. Samarati, "Access control: principle and practice," *IEEE communications magazine*, vol. 32, no. 9, pp. 40–48, 1994.

[2] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed nist standard for role-based access control," *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 3, pp. 224–274, 2001.

[3] N. K. Sharma and A. Joshi, "Representing attribute based access control policies in owl," in *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*. IEEE, 2016, pp. 333–336.

[4] T. Nguyen, S. Cuttill, T. Nguyen, and M. Mahdavi, "Identity and access management framework," Mar. 29 2007, uS Patent App. 11/731,011.

[5] T. Mather, S. Kumaraswamy, and S. Latif, *Cloud security and privacy: an enterprise perspective on risks and compliance*. " O'Reilly Media, Inc.", 2009.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[7] E. Stefanov, M. Van Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu, and S. Devadas, "Path oram: an extremely simple oblivious ram protocol," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 299–310.

[8] V. Narkhede, K. P. Joshi, T. Finin, S. Choi, A. Aviv, and D. S. Roche, "Managing cloud storage obliviously," in *IEEE International Conference on Cloud Computing (CLOUD), 2016*, vol. 2. IEEE, 2016, pp. 285–292.

[9] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," *Journal of the ACM (JACM)*, vol. 43, no. 3, pp. 431–473, 1996.

[10] I. Damgård, S. Meldgaard, and J. B. Nielsen, "Perfectly secure oblivious ram without random oracles," in *Theory of Cryptography Conference*. Springer, 2011, pp. 144–163.

[11] D. Boneh, D. Mazieres, and R. A. Popa, "Remote oblivious storage: Making oblivious ram practical," 2011.

[12] M. T. Goodrich, M. Mitzenmacher, O. Ohrimenko, and R. Tamassia, "Privacy-preserving group data access via stateless oblivious ram simulation," in *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 2012, pp. 157–167.

[13] E. Stefanov, E. Shi, and D. Song, "Towards practical oblivious ram," *arXiv preprint arXiv:1106.3652*, 2011.

[14] D. S. Roche, A. J. Aviv, and S. G. Choi, "A practical oblivious map data structure with secure deletion and history independence," *arXiv preprint arXiv:1505.07391*, 2015.

[15] M. Nabeel, E. Bertino, M. Kantarcioglu, and B. Thuraisingham, "Towards privacy preserving access control in the cloud," in *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2011 7th International Conference on*. IEEE, 2011, pp. 172–180.

[16] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 1, pp. 131–143, 2013.

[17] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[18] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering dac, mac and rbac," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2012, pp. 41–55.

[19] S. Godik, A. Anderson, B. Parducci, P. Humenn, and S. Vajjhala, "Oasis extensible access control 2 markup language (xacml) 3," Tech. rep., OASIS, Tech. Rep., 2002.

[20] K. Lalana, "Rei: A policy language for the me-centric project," *TechReport, HP Labs*, 2002.

[21] J. M. Bradshaw, A. Uszok, M. Breedy, L. Bunch, T. Eskridge, P. Feltovich, M. Johnson, J. Lott, and M. Vignati, "The kaos policy services framework," in *Proc. 8th Cyber Security and Information Intelligence Research Workshop*, 2013.

[22] T. Finin, A. Joshi, L. Kagal, J. Niu, R. Sandhu, W. Winsborough, and B. Thuraisingham, "R owl bac: representing role based access control in owl," in *Proceedings of the 13th ACM symposium on Access control models and technologies*. ACM, 2008, pp. 73–82.

[23] E. Yuan and J. Tong, "Attributed based access control (abac) for web services," in *IEEE International Conference on Web Services (ICWS'05)*. IEEE, 2005.

[24] H. Shen, "A semantic-aware attribute-based access control model for web services," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2009, pp. 693–703.

[25] J. Han, W. Susilo, Y. Mu, and J. Yan, "Attribute-based oblivious access control," *The Computer Journal*, vol. 55, no. 10, pp. 1202–1215, 2012.

[26] J. Camenisch, M. Dubovitskaya, R. R. Enderlein, and G. Neven, "Oblivious transfer with hidden access control from attribute-based encryption," in *International Conference on Security and Cryptography for Networks*. Springer, 2012, pp. 559–579.

[27] D. L. McGuinness, F. Van Harmelen *et al.*, "Owl web ontology language overview," *W3C recommendation*, vol. 10, no. 10, p. 2004, 2004.

[28] N. K. Sharma and A. Joshi, "Representing attribute based access control policies in owl," in *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, California, USA, Feb 2016, pp. 333–336.

[29] M. Li, S. Yu, K. Ren, and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2010, pp. 89–106.

[30] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," DTIC Document, Tech. Rep., 1973.

[31] K. J. Biba, "Integrity considerations for secure computer systems," DTIC Document, Tech. Rep., 1977.