A New Access Control Scheme for Protecting Distributed Cloud Services and Resources

Syed Rizvi and John Mitchell
Department of Information Sciences and Technology, Pennsylvania State University, Altoona-PA, USA
srizvi@psu.edu, jkm5270@psu.edu

Abstract
Cloud computing is clearly one of today’s most dominant paradigms in the Information Technology (IT) industry due to its scalable, flexible, and cost-efficient access to infrastructure and application services. Despite these promising facilities and benefits, cloud users have serious concerns about the data security and privacy. Among all security challenges, access control is of vital importance, since it provides security mechanisms to protect customer’s data, cloud services, applications, resources against unauthorized access, and misuse of privileges. Several access control systems have been proposed in literature lately but most of them are designed to work with one specific access control policy. In this paper, we present a novel generic access control scheme, capable to work with most of the available access control policies using a global resource management system (GRMS) to effectively handle both local and remote authentication requests. The introduction of GRMS makes our proposed architecture semi-distributed at the expense of minimal request-response time. We have shown the applicability of our proposed architecture using a case study for three different scenarios.

Keywords: Access control, cloud computing, side channel attack, role based access control, virtual resource manager.

1. INTRODUCTION
Cloud computing is clearly one of today’s most promising technologies due to its scalable, flexible, and cost-efficient access to computing resources. However, the increased concentration of business data and computing power scale security risks as well. This requires special considerations from service providers as cyber-attacks have been deployed to target cloud computing infrastructures. Defeating cyber-attacks in cloud computing environment is challenging due to both the scale of the problem and the increasing sophistication of attacks. One of the recent security concerns is attributed to the lack of a strong access control architecture that can protect cloud services and resources from unauthorized access and misuse of data.

Cloud computing supports multi-tenancy and virtualization within its infrastructure. Although it provides elasticity to customers through on-demand allocation and provides cost-effectiveness to service providers through an efficient resource allocation, it has its own security implications [26]. Multi-tenancy allows a single piece of software to be shared by many different users. Each user is referred to a tenant, and tenants can use the software as they like without being able to control the actual code of the application. Virtualization benefits allow for users to be able to use virtual copies of resources on the cloud with ease and efficiency. Since multi-tenancy and virtualization allow physical resources to be shared among potential untrusted tenants, this enables the cloud to be targeted by side channel attacks (e.g., time and bandwidth monitoring attacks) and the interference of tenant computation. In order to defend these attacks, it is crucial for the infrastructure to have distributed and structured access control architecture since properly secured authorization requirements are the only way to secure a cloud computing system.

To design a robust access control system, there are several requirements that must be addressed such as decentralized administration, secure distributed collaboration, credential federation, constraint specification, multitenancy, and resource sharing [1]. Each of these requirements is discussed below.

The first requirement of the decentralized administration is defined by an attribute of local autonomy [17]. A cloud environment is composed of several service layers where each service layer manages its own resources. To manage these resources, service layer enforces its own administrative rules that best match its security preferences and requirements. However, the cloud services are interoperable and shared among multiple users that demand a need for precise access control requirements that define what types of administrative rules the service models/layers can enforce such that services can flow between one another safely, securely, and efficiently. Therefore, to address this requirement, a decentralized administration is critical to enable each service model to have a local autonomy over its resources.
The second requirement is the secure distributed collaboration which is needed to support the decentralized administration of various service models, their resources, and access policies. Specifically, it must exist to allow both vertical and horizontal policy interoperation. As a result of the fluidity and independence of the cloud system, different service models may run by different resources and service usage policies. To address this requirement, it is important that all policies must be defined and verified by a service level agreement (SLA) to promote secure collaboration at different levels.

The third requirement is characterized as a credential federation which is needed to ensure the confidentiality of the user’s credentials that needs to be transmitted when a user attempts to obtain services from other connected clouds. The fourth requirement for a robust access control system is the constraint specification method that provides sufficient protection for services and resources in determining access for their entities [1] [2].

The fifth design requirement is the multi-tenancy aspect of the cloud that prevents a restrictive access control method such as MAC from being effective [3]. When this multi-tenancy combines with the distributed cloud service model, it brings several access control related challenges that demand a relatively generalized access control architecture that can operate with different service models within and across the cloud. To meet these challenges, our goal is to design an access control architecture that can adopt or work with any access control policy running at one specific cloud deployment model or service. To incorporate such flexibility in our proposed scheme, the component that holds different access control policies should be the integral part of access control architecture. Motivated with this, we introduce a policy bank (PB) as an integral part of our proposed architecture that can hold any desired access control policy. However, to show the practicality of our proposed scheme, we consider role based access control (RBAC) as an access control policy exists in the PB.

Finally, the last requirement is the resource sharing that allows users to share multiple cloud services that exist at different service layers within and across the connected clouds. Despite the obvious advantages of resource sharing, it brings a lot of challenges such as cloud service interoperability, service layer autonomy, routing of local and remote user’s requests for desired resources etc. For instance, we believe that well-designed access control architecture should provide some degree of autonomy to each service layer/model by which it can implement its desired access control policies with respect to its own requirements, current resources/services and other factors. To meet these challenges, one of our goals is to design an access control architecture that provides a great level of collaboration within and across these multiple clouds. To achieve this goal, we adopted a distributed access control policy that provides a complete local autonomy at each service layer (i.e., each service layer retains administrative control over its resources).

To address these six design requirements, the primary objective in this research work is to design semi-distributed access control architecture using three major components that work together: a global resource management system (GRMS), a virtual resource manager (VRM), and a peer access control module (PACM). To achieve our primary objective, we introduce the GRMS in our proposed architecture that serves as a barrier between different cloud services that may exist at the same or different layers. These resources/services are managed by a VRM which exists at each service layer with the sole purpose of managing virtual resources that are called for use. The proposed semi-distributed architecture preserves the autonomy of each service layer in applying the necessary access control policies and constraints. Because each layer has its own access control module, this allows each service layer to enforce its own access control policies and administrative rules with respect to its own requirements and preferences. This ensures that each local and remote request will be authenticated individually and the privileges or role(s) will be determined and assigned by the PACM using the access control policy stored in the PB. We refer our scheme as semi-distributed due to the fact that only one centralized GRMS is employed in our proposed architecture to avoid the use of a distributed SLA for each service layer.

This work is the continuation of our earlier research on access control for cloud computing paradigm [24]. Our proposed scheme differs from the existing access control schemes since it possesses the following four unique characteristics. First, the introduction of GRMS makes our proposed architecture semi-distributed at the expense of a minimal request-response time. Second, our proposed architecture works effectively with both PACM and VRM to protect and manage all resources and services of cloud providers from unauthorized access. Third, the centralized GRMS makes PACM completely autonomous in enforcing all the necessary access control policies to ensure the privacy of their customer’s data. Finally, the access control policies in our proposed architecture are not enforced across a single federate environment and therefore make PACM completely autonomous in policy enforcement.

The rest of the paper is organized as follows. Section 2 presents an analysis of current threats on access control in cloud computing. In Section 3, we describe the related work. Section 4 discusses the current access control schemes in cloud computing. In Section 5, we discuss our proposed distributed access control architecture. Section 6 presents a case study to show the practicality of our proposed scheme. Finally, we conclude the paper in Section 7.


Cloud computing presents many advantages to the cloud service user (CSU) and has become increasingly popular as a result. Many enterprises have migrated to the cloud in order
to capitalize on the virtually unlimited storage space, rapid data processing capabilities, and use-based pricing. However, since the cloud is responsible for the storage of important user data, it becomes an attractive target for attackers. There are several different attacks that can be launched against the cloud that violate its confidentiality, integrity, and availability. Specifically, an attacker may focus on the access control mechanism of cloud computing due to the vulnerabilities associated with multi-tenancy and virtualization. In this section, we examine the access control vulnerabilities of the cloud and the potential attacks aimed at the access control mechanism.

2.1 Issues with Access Control in Cloud.

Since the cloud is a dynamic and virtual environment, the traditional access control methods such as firewalls are ineffective against the wide variety of threats facing cloud computing [16]. In addition, the rapid fluctuations of virtual machines used for storing and analyzing data prevent current access control methods from being effective. Insider threats pose another serious risk and demonstrate the ineffectiveness of the current access control policies. For example, the Oregon Health and Science University was a victim of an insider attack that caused two data breaches and resulted in the exposure of sensitive medical information [16].

Furthermore, multi-tenancy and virtualization are two important characteristics of the cloud which are vital for its effectiveness. On the other hand, these features are also responsible for the security threats related to the access control mechanism. For instance, multi-tenancy promotes resource sharing among multiple users or tenants. These potentially untrusted tenants could perform side channel attacks and leak sensitive information pertaining to the cloud service provider (CSP) or the hypervisor hosting the targeted virtual machine [17] [18].

Moreover, the allocation of resources can allow for an attacker to easily infect other CSUs, since the same virtual resources are used. It provides a convenient delivery system that can be exploited and used by an attacker to infect the hypervisors and data centers. An insecure API represents another potential avenue for an attacker. Access control in the cloud is dependent on the API which is used for multiple purposes such as encryption and authentication [19]. The design of the interfaces must be secure enough to withstand exploitation.

2.2 Multitenancy and Virtualization.

Multi-tenancy is the root cause of several access control challenges that are specific to cloud computing. As previously mentioned, resource sharing facilitates the possibility of attacks whereas interference among the tenants could cause unauthorized information flow. The isolation between the virtual machines must be increased to ensure an efficient access control mechanism for cloud computing [20] [21].

In addition, many of the access control threats in cloud computing focus on exploiting the virtualization aspect of the cloud in order to steal user data or manipulate resources. The hypervisor is a common target for attackers since it responsibility for managing the guest virtual machines. If an attacker can compromise the hypervisor, further attacks can be launched that affect the hosted guest virtual machines along with the resources shared between them. These attacks include: theft-of-service, malware-injection attack, side channel attacks, distributed denial-of-service (DDoS) attacks, and rollback attacks.

2.3 Side Channel Attacks.

Side channel attacks represent the most significant threat to the access control mechanism of the cloud. Instead of directly focusing on the cryptographic algorithms, the side channel attacks incorporate the physical hardware and implementation of the system. A side channel attack establishes a communication channel between the tenants that utilize the shared resources of the cloud. This allows the attacker to transfer sensitive information such as identification photos or banking credentials across many different virtual machines [21]. An attacker can launch a side channel attack by taking advantage of the execution times of virtual machines overlapping with each other. In addition, the side channel attack is difficult to prevent in a cloud environment since sharing any type of resources can create a communication channel.

Since side channel information can be acquired using a variety of methods, there are many classifications of side channel attacks. The timed side channel attack is a variation of the original attack which involves the malicious user inferring the behavior of the targeted virtual machine in order to determine the computation time and other important information. The goal of this attack is to find the private key by monitoring the cryptographic operations being performed. To weaken the timed side channel attack, the CSP should employ noise injection before and after the cryptographic operations [22]. However, this preemptive measure will not completely prevent the attacker from discovering information that could be used to find the private key. In order to defeat the timed side channel attack, the encryption times must execute simultaneously. The CSP can equalize the encryption times if branch processing is removed from the cryptographic algorithm. Unfortunately, this will increase the cost substantially which circumvents the advantages provided by the cloud.

Another type of side channel attack is the cache based attack which is similar to the original attack. The main difference is that the cache is used instead of the main memory to transfer data [18] [21]. This attack aims to measure the delay incurred when the CPU loads the selected data from the main memory to the cache. The cache based
side channel attack has broken several reliable software ciphers such as DES and AES [22]. The most common countermeasure for preventing cache based attacks is to remove the cache from the system.

Fault side channel attacks present another significant threat to the CSU since it is more effective than most of the other types of attacks that target the access control mechanism. If the CSP is unreliable, there is an increased possibility of hardware faults. For instance, an untrustworthy CSP may not disclose past security incidents or undertake the efforts to improve upon the security infrastructure. These faults can provide a communication channel that has a higher probability to be exploited to launch additional attacks. Moreover, an attacker can attempt to create a communication channel by sending corrupted input data and receiving an error message from the targeted CSP [22]. Once the channel has been established, the attacker can begin exploiting the fault. In general, the attacker will utilize cryptanalysis to deduce the private key. One of the known countermeasures for mitigating the fault side channel attack is to allow for two transformation rounds, since an attacker is unable to cause the same fault more than once [22]. However, the additional transformation round will cause significant performance degradation as a result.

Power consumption is a physical characteristic that can potentially provide an attacker with vital information related to the system implementation. The power analysis attack exploits the information related to the power consumption in order to decrypt cipher-text and uncover the private key. This attack can be further categorized into simple power analysis (SPA) and differential power analysis (DPA) [22]. In regards to cloud computing, the most effective type of power analysis is DPA, since SPA requires the attacker to have complete knowledge pertaining to the physical implementation. Unless the attacker is a malicious insider, the DPA will be the preferred method of attack. It is also important to note that the DPA demands fewer resources than most of the other side channel attacks. Statistical methods are employed in a DPA attack which enables the attacker to compare two traces and determine the correlation. If there is a correlation, the power consumption will exhibit sporadic changes which indicate that the correct secret bits were used [22]. Most of the countermeasures aimed at defeating the DPA attack only reduce the power signal. Therefore, there is no effective countermeasure for preventing the DPA attack.

2.4 Theft of Service Attack.

The theft-of-service attack is performed when the attacker manipulates the scheduling mechanism of the hypervisor to allow for the unmonitored use of resources [18]. Specifically, the periodic check performed by the hypervisor is unable to detect the attacker. This attack is more harmful when performed against a public cloud, since the cost associated with resource use would be nonexistent for the attacker. In addition, a greedy attacker could utilize the resources for any duration of time regardless of the priorities of other CSUs. As a countermeasure, the CSP could modify the hypervisor to promote a higher level of fairness and detectability.

2.5 Malware Injection Attack.

The malware injection attack occurs when an attacker injects malicious code into the cloud in order to deceive the hypervisor. This malicious code will create a copy of an instance which appears to be legitimate and allowed to operate using the cloud services. Eavesdropping becomes one of the most critical problems that are caused by the malware injection attack [23]. Furthermore, if the malicious instance is a copy of another CSU instance, it will intercept some of the services requested by the victim. The security issues caused by the malware injection attack include user data leakage and unauthorized access to the resources of the cloud [18]. The examination of the memory is one of the countermeasures used to detect malicious activity. There are not many countermeasures that prevent malware injection attacks due to the fact that the hypervisor does not check the integrity of the malicious instance [23]. Rather, it only checks if the malicious instance matches one of the legitimate instances.

2.6 Virtual Machine Rollback Attack.

A virtual machine rollback attack is another example of an attack utilized by a malicious user that exploits the hypervisor vulnerabilities. This attack occurs when a malicious user obtains a virtual machine snapshot and erases the history to prevent the hypervisor from identifying abnormal behaviors. It allows the attacker to perform brute force attacks indefinitely since the attacker can erase the history before the restriction on the number of attempts is reached [18]. Since the hypervisor periodically takes a snapshot of guest virtual machines without the knowledge of the user, there are multiple snapshots that can be exploited by the attacker. One of the countermeasures is to limit the functionality of the hypervisor. In particular, the ability for a hypervisor to suspend and resume the virtual machines should be disabled in order to prevent virtual machine rollback attacks. However, the hypervisor is essential for managing virtualized environments and suspend and resume feature would also remove many of the advantages provided by the virtualization aspect of the cloud [18].

3. Related Work

There are several access control policies and authorization mechanisms that have been proposed in literature in order to defend against cloud security threats. Almutairi et al. [1] propose a distributed access control architecture which is based on the RBAC model. The proposed architecture is designed to mitigate and prevent an attacker from exploiting the virtualization or multi-tenancy.
features of the cloud. This is one of the closest works to ours since the three components included in their approach are identical to those utilized in our own architecture. It is also important to note that the proposed architecture is capable of supporting multiple access control policies besides the RBAC model since the design of the architecture is generic.

In addition, the authors identify several metrics and collaborations that allow for an efficient distributed architecture.

Li et al. [3] propose the "New Advance RBAC Architecture" system. They discuss that the current RBAC system has a few drawbacks such as the lack of restrictions that prevent a certain number of users from being assigned to the same role. This feature is included in the proposed architecture since there is a limit on the amount of roles generated which allows an admin to easily identify unauthorized users. Moreover, the proposed architecture has capabilities that allow it to minimize the possibility of hijacking a user id or accessing important data through a fake id by limiting the amount of potential users.

In [4], the authors propose a risk-based dynamic access control model to address the access management difficulties associated with cloud computing. This model uses three new components which are as follows: the risk engine, the risk policies, and the risk quantification web services. The authors believe that previously determined access decisions are not as effective as dynamic access models since unexpected situations may occur in real life. These situations require a flexible policy in order to promote the most efficient access decisions, and the authors believe that contextual information should be included in the access control model for the cloud. Furthermore, the characteristics of the model presented by the authors would allow for flexibility while maintaining the security of the cloud. However, there are several issues that prevent a risk-based access control model from reaching its potential. The process or method used to make decisions such as decision theory and operating the model in real-time limit the effectiveness. The authors also claim that performance degradation and overhead are two serious considerations when using the proposed model.

Thomas et al. discuss the advantages and disadvantages of distributed access control and the configurations presented by other researchers [5]. The authors believe that an agent based approach is the most efficient method for mediating the access requests of users. They present several advantages of an agent based approach which include: dynamic adaptability, robustness of the system, and the ability to operate in different modes. The authors claim that one of the main drawbacks of an agent based approach is the lack of a management area in the cloud that deals with a conflict in policies. To improve upon existing approaches, the authors incorporate a policy conflict manager which enforces the security policies.

In [6], the authors propose a framework that mitigates the cloud security threats by enforcing the organizational policies. In addition, the authors describe the current access control models and present the merits and demerits of each one. They claim that a RBAC model is superior to previous models since it offers better management when assigning permissions to the users. To promote the highest level of efficiency from their proposed framework, the authors designed a RBAC model that runs as an additional layer on top of the Open Nebula framework. In similarity to other works, the authors believe that contextual information is critical in order to achieve optimal performance of the system.

Chow et al. [7] discuss the growing popularity of the cloud and the lack of trust which prevents many individuals from migrating to the cloud. They believe that outsourcing important information and processing sensitive data is not supported by the existing access control frameworks. Moreover, they argue that cryptographic protocols are necessary to protect the confidentiality and integrity of the user’s data. The authors claim that their proposed approach can employ encryption techniques which do not hinder the data processing advantages of the cloud while promoting security. This is achieved by limiting the control of the cloud service provider.

Amit et al. [8] identify and describe the main cloud security threats which prevent the widespread adoption of the cloud. One of the most important issues that the authors identify is the lack of trust associated with cloud computing. They propose an approach that leverages the ideas presented in traditional trust models to mitigate the existing security issues and develop a more efficient trust management system.

Berger et al. [9] propose an authorization mechanism using trusted virtual data centers (TVDc) that improves upon a previous work. They believe that access to cloud storage should be limited and controlled through isolation and constraints. The authors use security labels to provide authorization for accessing the cloud storage and virtual and physical resources.

Zhang et al. [10] propose a framework for securely developing elastic applications for the cloud. In addition, the framework increases the performance of the elastic applications and exploits the flexibility of the cloud by allocating resources based on demand. In regards to authorization procedures, the authors believe that applications should incorporate the concept of least privileges, since the permissions assigned to each application might change based on the location of the execution.

4. CURRENT ACCESS CONTROL SCHEMES IN CLOUD COMPUTING

Access control has been one of the most critical issues in cloud computing since the cloud shares physical resources and applications with various organizations and users. However, if an IT department disables access strictly for security reasons, the cloud system will lose its shared service
purpos. Therefore, flexibility is a necessity when designing the access control mechanism since it must be able to accommodate different types of policies and domains. There are many methods designed to prevent an intruder from accessing a cloud computing system; Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role Based Access Control (RBAC), Attribute Based Access Control (ABAC) are four major access control methods used for various cloud infrastructures. Table 1 compares the four access control methods and provides an examination of the benefits and drawbacks of each method. We also explain the aforementioned types of access control which are already used in the cloud computing system and the different characteristics they contain.

### 4.1 Discretionary Access Control (DAC)

DAC is an access policy where the owner has complete control and determines the authorization and permissions of other users. In addition, once one user has ownership of an object, permissions may be granted to other necessary users. DAC is typically used with an access control list (ACL). Thus, DAC makes it possible for the owner to easily and conveniently set up user privilege. This approach allows the user to easily access the authorization database which contains the index of authorized users whenever a modification is required. However, DAC is not able to ensure the flow of information for those that request and restrict the usage of information for those that are not authorized. This problem can become a potential vulnerability that can be exploited, since an attacker could threaten data integrity [3].

### 4.2 Mandatory Access Control (MAC)

MAC is an access policy where the administrator is responsible for creating the policy. On the other hand, DAC is determined by the owner. Since MAC is more restrictive than DAC, MAC is used by organizations that contain important data or classified information [11]. The users or owners of an object are unable to grant authorizations or permissions to others. The administrator is the only individual that has the ability to modify the security status of another user. MAC is used to protect networks, file systems, and enforce user authorization by preventing unauthorized users from accessing private information. A MAC label may be applied to a user which will limit the amount of decisions available to the user [12]. Moreover, when the security level of an individual or subject requires modification, it will be impossible to change the security level. This is one of the major limitations in MAC.

### 4.3 Attribute Based Access Control (ABAC)

In recent years, the ABAC has become increasingly significant due to the growing popularity of large distributed systems [15]. One of the main drawbacks of RBAC is the difficulty of assigning privileges to an individual. ABAC provides an effective solution since user attributes are the criteria used to determine user authorization. This access policy improves upon RBAC in the following areas: delegation of attribute authority, decentralization of attributes, and interference of attributes [12]. To protect the sensitivity of credentials, ABAC contains several policies for maintaining user confidentiality and data integrity. ABAC is highly flexible and capable of supporting a large variety of policies and domains. In addition, ABAC has the

| Table 1. Comparative analysis of access control methods |
|---|---|---|---|
| **Access Control Methods** | **MAC** | **DAC** | **RBAC** | **ABAC** |
| **Restrictiveness** | High | Low | Medium | Medium |
| **Flexibility** | Low | High | Medium | Medium |
| **Policy Maker** | System | Owner | Roles | Attributes |
| **Advantages** | Highly secure | Easy privilege configuration | Supports large enterprises; mitigates damage to data | Requires investigation for defining attributes relevant to authorization decision |
| **Disadvantages** | Low storage capacity; unable to modify security levels | Low storage capacity; unauthorized privileges can be granted | Privileges depend on role | |
capability to provide an automated trust negotiation, which allows for auditability when necessary [15].

4.4 Role Based Access Control (RBAC).

RBAC is an access policy that is formulated from the idea that a role should provide permissions to an authorized individual. RBAC is optimized for enterprises and large-scale applications and also supported by a commercial DBMS (Database Management System) [13]. Additionally, RBAC is encouraged by organizations that want to manage the authorization of users conveniently and apply access controls with their policies. In order to understand the RBAC process, it is important to examine and define the three primary rules used by RBAC [3]. The first rule is role assignment which states that a role must be assigned before an individual can exercise any permission. The second rule is role authorization which requires an individual to be authorized for the role that is assigned. The third rule is permission authorization which prevents an individual from exercising unauthorized permissions by assigning each role its own permissions.

RBAC provides access to a user based on role hierarchy which is determined by the number of applications. Fig. 1 displays the format of a RBAC policy where different users in an organization are assigned to different roles that determine the privileges of each user. The concept of least privilege is used to determine role assignment for a specific object in order to mitigate damage caused by possible attacks.

To prevent the misuse of information, there is a separation of roles which refers to an individual role being assigned to each user. RBAC follows a certain array of administrative policies in order to work efficiently which are classified as: centralized, hierarchical, cooperative, ownership, and decentralized [3]. Some researchers believe that RBAC should be employed as an access policy in cloud computing. However, it is sometimes difficult to determine which privileges belong to an assortment of different users, and which users receive the various different kinds and levels of roles. The privilege of role change allows the permissions assigned to each of the roles to be modified. When attempting to change the role of a user, it may generate confusion since each job role has its own set of permissions. Moreover, since the cloud is a relatively new paradigm, there are still several issues that need to be resolved in order for RBAC to achieve its full potential and provide a more efficient access control [14].

5. Proposed Distributed Access Control Architecture

Once appropriate authorization requirements have been met, a distributed cloud security architecture can be designed with the ultimate goal of secure and reliable access control. This architecture can be built using three major components that work together: a VRM, a PACM, and a GRMS. To show the practicality of the proposed approach, we assume a RBAC format as an underlying access control architecture due to its straight-forward administration and its ability to scale up and down with ease. However, the design of the distributed security architecture is flexible and generic enough to work with other types of access control policies such as discretionary access control. Access to a cloud-computing infrastructure begins with a simple request from a user. Before we can take a look at the three components that make up the distributed access control architecture, we must take a deeper look at the flow of the cloud user requests that the system will handle.

Based on the RBAC policy, the cloud system user request (CSUR) can be represented using four-tuple expression as follows: user credentials, privileges, requested resource/service, and an activity log file (see Fig. 2). Every request begins with a user first authenticating through an access control module. Upon validation of the user’s credentials, user privileges are determined and assigned to the client based on the user role(s). The user’s privileges will be visible and processed by all the PACMs receiving requests to make sure that users do not enter unauthorized areas of the cloud infrastructure. The requested resource section of the CSUR will act as the body of the request for the VRM to read and provide available resource to the requested users. The activity log file will keep track of all access activity that occurs under a user’s credentials and assigned privileges. This information will then be sent to a log file database linked to the user credential database at the backend of the cloud computing infrastructure. This will be important when investigating malicious incidents and activities that occur within the cloud computing system. The
authentication and authorization processes using the PACM are shown in Fig. 3.

The following are some of the significance of our proposed architecture.

a. Autonomy of PACM in enforcing the access control policies (i.e., PACM serves as a first layer of defense).

b. Allows both intra and inter cloud communication to achieve maximum collaboration.

c. The way the GRMS is connected to each access control module, the remote requests cannot traverse more than 2 hops (minimum request-response time).

d. Log file is created that can be used to feed intrusion detection and prevention system (2nd layer of defense).

e. Log file can also be used by GRMS to ensure “compliance” of each cloud provider in providing services to the customers.

5.1 Peer Access Control Module (PACM).

In our proposed architecture, a PACM is envisioned as a gateway that exists on each service level of a cloud to protect the resources from any direct unauthorized access. It works by focusing on processing and approving authentication and authorization requests that come in and out of the cloud before reaching the VRM for the deployment of services. Also, each access control module is directly peered with the GRMS to entertain remote service requests. In addition, it provides an activity log file of each authenticated user to GRMS, so that the current resource occupation and the service utilization of each cloud level can be maintained and monitored. The GRMS can also use this information to detect any malicious activities at the global virtual level. Thus, the GRMS will have the most accurate information about all of the available resources managed by the VRM. This enables our proposed architecture to ensure that the remote service requests will not be forwarded to more than two hops which reduces the overall request-response time.

Furthermore, the PACM can be considered as a processing-center which is made up of the following parts: an access control gateway, a credential extractor/evaluator, an activity log file, and a policy bank (PB). When a request comes in, it enters the access control gateway. At this point, the request contains the client information, access credentials, the desired service from the cloud level, and the desired level of privileges and permissions of that service. Fig. 3 demonstrates this authentication and authorization process as well as the six steps that are involved before the deployment of the desired services or resources.

When a user submits a request to the PACM, it comes with the desired authorization level and the specific authentication credentials that will be used throughout the request process. These credentials are extracted by the access control gateway and sent to a credential evaluator for the authentication purpose (step 2). The credential evaluator checks the user credentials against the local database and makes the access decision (step 4). If the request is denied due to incorrect credentials, the client will be notified through the access control gateway. However, if access is granted, the request will be forwarded to the ADP (step 5). The ADP forwards the request to the role assigner which evaluates the desired privileges (e.g., read or write permission) for the requested service/resource based on the user-role of the requesting client (step 6). In our case, this role assessment will be evaluated and determined based on the stored RBAC policy in the PB (step 7). Once the authorization levels are determined through the PB, the approved request with the level of authorization will be sent to an appropriate VRM (step 8). Constraints can be applied based on the request by the role assigner with respect to its assessment of the requested service and the corresponding requested privileges found in the user request. This will ensure that users are only able to use the services and resources within the context of their request.

Once PACM approves a client request, the credentials should be stored in the local cache of the PACM to ensure that the future requests from the same client will not have to go through from all the regular access control procedures. This will speed up the authentication process at the expense of the creation of a local cache.
of relatively small storage usage. However, the authorization level needs to be determined and assigned by the PACM each time based on what services or resources were requested by the client. This will ensure the strict enforcement of the access control policy on all of the services and resources the cloud provider offers to its clients.

Once the approved request reaches the VRM, it will check the availability of the desired resource. The structure of the approved request is shown in Fig 4. If the requested resource is available, it will be deployed locally to the client and the activity log will be initiated by the PACM. If the requested resource is not available or it does not locally exist at the cloud level, the user request will be sent to the GRMS for further assistance. The GRMS will then locate the requested resource and forward the user request to the appropriate PACM. As a result, the PACM will remain autonomous in terms of applying its local access control policies on the user request rather than a mediated policy.

5.2 Virtual Resource Manager (VRM).

Cloud computing is a method of delivering computing services from a large, highly virtualized data center to many independent end-users, using shared applications and pooled resources [25]. Due to heterogeneity and granularity of virtual resources, it is desirable to have a VRM available at each service layer of a cloud with the sole purpose of managing virtual resources that are called for use. It is responsible for deploying services and calling them back when they are no longer in use or the request is timed out for that particular resource. Whenever the requested resource does not exist or if the resource is not available locally, the request will be considered as a remote service request and forwarded to the GRMS. The GRMS will use its global database to forward the remote request to an appropriate VRM through the destination cloud PACM. The destination PACM can apply its local access control policies to ensure that only the authorized and requested services are drawn for usage.

5.3 Global Resource Management System.

A GRMS acts as a barrier between different cloud services on the same layer or different layers. Fig. 5 shows the significance of the GRMS as a mediator between multiple cloud service types. The use of one centralized GRMS in our proposed architecture is to avoid the use of an SLA for each service layer (e.g., SaaS, IaaS, and PaaS) as proposed in [1]. If the SLAs of each layer on different cloud platforms are not tightly coupled, the remote service request

![Figure 3. Authorization Request Process for Deployment of Services through PACM](image-url)
may go into a deadlock situation when searching for the right VRM. It also does not guarantee how many hops the remote service request needs to traverse before it reaches the desired VRM. Therefore, the use of a centralized GRMS, peered with each PACM, not only simplifies the access control design but also reduces the overall request-response time.

To avoid single points of failure of the GRMS, redundancy can be introduced. Although not required in our proposed architecture, a single SLA can reside in the GRMS to overlook all of the standard policies and procedures so that compliance as well as QoS can be ensured at the global virtual level. When a user requests resources from a local cloud, but they do not exist in the specified location, the request can be processed through the centralized GRMS. However, if the desired resource exists within the same cloud but at a different layer, the request can be sent directly to another cloud service sector or layer in order to remotely retrieve the needed service and resource.

6. Verification Of The Proposed Scheme

In this section, we present the use cases to show the practicality of our proposed access control scheme. The use cases illustrate the interaction between all elements of our proposed architecture such as VRM, PACM, GRMS, and CSUR. Specifically, it shows different aspects of our proposed scheme such as the user authentication and authorization, role assignment and policy enforcement, and resource deployment and management. To show all these characteristics, we present a single case study with three possible scenarios. In the first scenario, a cloud user is requesting for a resource or a service that exists at the first PACM. In the second scenario, we assume that the requested resource or a service exists within the local cloud but at a different layer. Finally, in the third scenario, the requested resource or a service does not locally exist and needs to be dealt at the global virtual level by the GRMS. For each scenario, we show what steps should be taken by our proposed architecture to (a) enforce authentication and authorization at the PACM (b) perform role assignment, resource type finding etc. at the ADP level and (c) provide requested services/resources to legitimate users by either local VRM or GRMS.

6.1 Initialization Phase.

Fig. 6 and Fig. 7 illustrate three classes of scenarios covering all possible interactions within and across multiple clouds. These scenarios involve the three types of communication discussed earlier in this article. Initially, when a user request arrives at one of the cloud providers, it will be received by the PACM, as shown in Fig. 6. Specifically, the CSUR (X; isolation constraint; app; LFxi) will be originated by the requesting user and arrived at the access gateway of the local PACM. The CSUR can be interpreted as follows: User X requests for a desired service (e.g., virtual resource or an application) to perform certain operation(s) (e.g., execution, update, etc.) with respect to its role-based authorization-level (e.g., isolation constraint) and bounded by the activity log-file.

The local PACM forwards the CSUR to the credential extractor (CEX). The credential extractor retrieves the user’s credentials from the CSUR (e.g., authentication credentials, context information, etc.) in order to perform authentication process. The extracted information will be forwarded to the credential evaluator (CEV). The credential evaluator checks the user authentication information (such as username, password, token, etc.) against its protected local database. If the credential evaluator grants the request (i.e., CSUR), it routes the request to the ADP to determine the role and the respective privileges.
The ADP consists of a role assigner (RA) and a policy bank (PB). The ADP receives the CSUR along with the authorization credentials. The RA determines the role of the requested user based on the type of the requested resource or service and its availability within the cloud. The resource/service-type indicates what layer the desired resource exists and what privileges can be assigned for that particular resource/service. The privileges and other access control policies will be enforced at the time of authorization using the local PB. Based on the type of the requested resource (e.g., SaaS, PaaS, and IaaS), there can be three possible scenarios.

6.2 Scenario 1 – Intra-Layer Communication.

In the first scenario, we assume that the requested resource or service exists on the same layer of the local cloud. For instance, user X initiates the authentication process with the PACM($\text{SaaS}$) to access an application at the SaaS layer (e.g., $\text{App}_{\text{SaaS}}$) of its own cloud. In this case, once user X is authenticated, the user-role will be determined based on the user credentials (for example, $R_x$) and the service_type will be assigned as ‘Local-Request’ by the RA. The RA notifies the ADP about user X request such as: \{X, $\text{App}_{\text{SaaS}}, R_x, \text{VRM}_1\}$. In response to RA feedback, the ADP will issue the Approved - Cloud system user request (A-CSUR). Moreover, the ADP will send the A-CSUR to the corresponding VRM. The local VRM ($\text{VRM}_1$) of the SaaS layer will deploy the requested resource/service according to the authorization-level(s) specified in the A-CSUR. At the same time, the designated VRM updates a local log-file for user X for its CSUR. The local log file can be shared or synchronized with the log file maintained by the GRMS. This log information can be used to deal with the malicious users or activities at a later time.

6.3 Scenario 2 – Intra-Cloud Communication.

In the second scenario, we assume that the desired resource or service resides in the local cloud but at a different level. In this case, the user request will be forwarded to the local PACM of the other layer (i.e., the layer which contains the requested resource/service). For instance, user X wants to access a storage (say ST) resource at the SaaS layer (e.g., $\text{ST}_{\text{SaaS}}$). Let’s assume that user X request arrives at the Layer 2 of the same cloud. The request will be authenticated (e.g., using $CE_X$ and $CE_{ei}$) and authorized (using the RA and the PB) by the local PACM to determine which VRM should be called. Since the type of the requested resource belongs to SaaS, the ADP will forward the request to the local PACM of Layer 1 along with the authorization information. This will be done through intra-cloud communication where each PACM is connected to the other PACM to exchange local requests.

Although not required, the PACM($\text{SaaS}$) of Layer 1 can send the request to the RA to determine which access control policies may need to be applied before calling the local VRMS. Once the desired policies are enforced and the authorization privileges are determined, the local VRMS will allocate the desired service or resource to user X. Finally, the designated VRM updates a local log-file for user X for its CSUR.
6.4 SCENARIO 3 – INTER-CLOUD COMMUNICATION.

This is a scenario where a requested resource does not exist locally as shown in Fig. 7. Therefore, the requesting client should be authenticated by the first PACM. However, the request should be forwarded to the GRMS, so that the appropriate VRM should be selected from another service provider. In this case, the GRMS selects the cloud provider and sends the request to the corresponding PACM. Although, the client has already authenticated by the first PACM, the second PACM would be independent in applying all necessary local access control policies on the user X request. This ensures the autonomy of each cloud provider in enforcing its own local access control policies and therefore minimizing all potential threats on its customer’s data. This is a very important aspect of our proposed architecture since each cloud provider has its own threat environment which demands separate customized access control policies based on the sensitivity of their data and resources. For example, a cloud provider which holds extremely sensitive information of their clients would likely implement strict access control policies on each access request as compared to a cloud provider that holds relatively less sensitive information. Since each cloud provider is connected to others through GRMS, they should be completely independent in applying their access control policies.

7. CONCLUSION

Cloud computing is a household name, and the virtual and scalable services it provides makes it almost critical to the success of any modern day business that works with IT solutions. As a result of its criticality, it is extremely important that the security of such a system is taken very seriously. Access control within a cloud infrastructure has always been a matter of contention. Cloud computing allows multiple tenants to access and request services at the same time virtually, but this can leave an insecurely designed cloud infrastructure quite vulnerable to problems such as side channeling and interference attacks. Due to this trade-off of openness to resource disruption, a problem arises: how can we design the access control architecture in a way to defend against side channeling and interference attacks? We discussed the related works of others who have tried to solve
this problem, and each method had its own pros and cons. For this paper, however, we discussed the security of the cloud infrastructure while focusing primarily on distributed access control security architecture. Within this system, the following authorization requirements are essential before designing the distributed access control architecture: a decentralized administration, a secure distributed collaboration, and a credential federation. Once this has been established, we discussed that when designing a distributed access control architecture, it will have three major parts that work together to process access-requests: a PACM that accepts/denies/reroutes access requests, a VRMS that deploys and monitors resources and services, and a GRMS that handles the movement of requests to other clouds for remote service/resource usage. How does this tie back to our problem? The access control module will make sure that users are only able to use the services and resources within the context of their request. Moreover, the policy bank contains an access policy that conducts role mapping for user requests and defines specific constraints for sharing resources among multiple users. Ultimately, these constraints will eliminate the possibility of side channel attacks. To show the practicality of our proposed access control architecture, we presented a case study using three different scenarios. For each scenario, we showed what steps should be taken to process both local and remote access-requests.

8. REFERENCES


Authors

Syed Rizvi is an Assistant Professor of Information Sciences and Technology at the Pennsylvania State University - Altoona. He received his doctorate in Modeling and Simulation from the University of Bridgeport in 2010. His research interests lie at the intersection of computer networking, information security and modeling and simulation. Recently, he has been working on security issues in cloud computing, cognitive radios for wireless communications, and modeling and simulation of large-scale networks. His expertise includes the design, analysis, implementation, optimization, and comparisons of algorithms in the areas of wireless/multiuser communications, information security, and parallel/distributed systems. He has authored and coauthored several technical refereed and non-refereed papers in various international conferences, journal articles, and book chapters in research and pedagogical techniques. He is a member of IEEE Communications Society and the ACM.

John Mitchell is currently an undergraduate student in the Department of Information Sciences and Technology at Pennsylvania State University. Recently, his research has involved the security issues of biometric identification systems with cloud computing, and virtualization techniques for improved biometric performance. He is currently pursuing a bachelor’s degree in Security and Risk Analysis.