Security Architecture for Cloud Computing Platform

SANJAYA DAHAL

Master of Science Thesis
Stockholm, Sweden 2012
TRITA-ICT-EX-2012:291
Abstract

Cloud computing is an innovation of existing technology which provides long-dreamed vision of computing as utility. The emergence of this novel technology in IT business has decoyed most of organizations in both private and public sector. Although cloud introduces the innovative and cost effective concept of on demand service, pay as you go, and resource allocation, security is often the area of concern in terms of its adoption. The existing security-based solutions for cloud-based platform are either based on single tamper-proof hardware or homomorphic encryption. Hardware-based solution lacks scalability, while homomorphic encryptions are only a theory. Moreover, traditional defense in-depth security mechanism cannot be directly implemented in cloud-based platform due to the varying nature of its service and deployment model. However, the same concept of multi-layered security mechanism can be proposed to secure the cloud-based platform.

This Master Thesis research is focused on deriving the generic and secure architecture for cloud computing platform regardless of its services and deployment model. The research focus on delivering seamless access control, authorization, identity and SSO services to end-user. All of the above mentioned services are offered by the components of our central security system. The central security system is the purposed architecture for cloud computing platform, which is based on service oriented architecture where all the security services are provided in terms of web services to end-user. Finally, OpenStack being an open source cloud computing platform is selected as a targeted platform in order to deploy and evaluate security services offered by our central security system.
Acknowledgement

I am honored to work with my supervisor, Professor Sead Muftic. I will also like to thank him for his guidance and support during this thesis work.

I will like to express my gratitude to whole SecLab team for their collaborative effort and technical support. Finally, I will like to thank to my whole family for their invaluable support and motivation during this entire period.
# Contents

Abstract 3  
Acknowledgement 4  
List of Figures 8  
List of Tables 9  
Abbreviation 10  
Chapter 1 11  
  Introduction 11  
    1.1. Background 12  
    1.2. Problem Statement 13  
    1.3. Purpose and Goals 15  
    1.4 Research Methodology 15  
    1.5 Thesis Outline 16  
Chapter 2 17  
  Security Issue in Cloud Computing Platforms 17  
    2.1 Background 17  
    2.2 Security Issues Identified By Cloud Security Alliance 17  
    2.3 Security Issues Identified by NIST 19  
    2.4 Security Issue Identified by ENISA: 21  
Chapter 3 23  
  Overview of Security Mechanisms for Cloud Computing Platform 23  
    3.1 Background 23  
    3.2 Security Overview 23  
    3.2 Security Mechanisms for the Cloud Service Model 24  
Chapter 4 27  
  Design of the Cloud Central Security System 27  
    4.1 Background 27  
    4.2 Security System Architecture 27  
    4.3 Cloud Security Infrastructure 28  
    4.4 Security Services offered by our Central Security System 29
List of Figures

Figure 1: Cloud Architecture ................................................................. 13
Figure 2: Central Security System Architecture ........................................... 28
Figure 3: Cloud Security Infrastructure ....................................................... 29
Figure 4: OpenStack Compute Structure ................................................... 34
Figure 5: OpenStack Glance Structure ....................................................... 35
Figure 7: Swift Structure ........................................................................... 36
Figure 8: Keystone Structure ..................................................................... 37
Figure 9: OpenStack Structure with Horizon Dashboard ............................... 38
Figure 10: OpenStack Deployment Strategy ................................................ 40
Figure 11: Authentication Mechanism with Keystone ................................. 41
Figure 12: Initial Authentication in Keystone ............................................... 42
Figure 13: Authentication and Delegation Protocol ...................................... 43
Figure 14: Integration of IDMS with Keystone ............................................ 45
Figure 15: IDMS server User Registration Form ......................................... 48
Figure 16: Authentication using One-Time Password in HTTPS .................... 49
Figure 17: An Encrypted and Signed Token Provided by Keystone in PKI mode 49
Figure 18: User Role Registration ............................................................... 50
List of Tables

Table 1 Security issue identified by CSA.................................................................18
Table 2 Security issue identified by NIST..................................................................20
Table 3 Security issue identified by ENISA..............................................................21
Table 4 System Security evaluation........................................................................52
Abbreviation

NIST National Institute of Standards and Technology
CSA Cloud Security Alliance
ENISA European Network and Information Security Agency
FIPS Federal Information Processing Standards
JSON JavaScript Object Notation
IaaS Infrastructure as a Service
PaaS Platform as a Service
SaaS Software as a Service
SAML Security Assertion Markup Language
XACML extensible Access Control Markup Language
PDP Policy decision Point.
PEP Policy Enforcement Point
LCA Local Certification-Authority
CSP Cloud Service Provider
SOA Service Oriented Architecture
CMS Cryptographic Message Syntax
Chapter 1

Introduction
Cloud computing is the evolution of an existing IT infrastructure that provides a long-dreamed vision of computing as a utility. The emergence of cloud technologies over the last several years had significant impacts on many aspects of IT business. According to the survey conducted about cloud computing, most of medium and small companies use cloud computing services due to various reasons, which include reduction of cost in infrastructure and fast access to their application [2]. Cloud computing has been described in terms of its delivery and deployment models. Although cloud computing emerges from existing technologies, its computing (delivery and deployment) models and characteristics raise new security challenges due to some incompatibility issues with existing security solutions [4].

Security is often an area of concern for both cloud vendors and consumers. Hence, it represents an urgent priority [3] in the market of IT business. According to the survey conducted by International Data Corporation (IDC), Microsoft [10] and NIST [5], security in a cloud computing model was of primary concern for transformed IT executives [9][10]. Researchers from MIT believe that “Information technology's next grand challenge will be to secure the cloud” [11]. National Institute of Standards and Technology (NIST) [5] also points "security challenges of cloud computing presents are formidable". Kui Ren, Cong Wang, and Qian Wang in [1] discuss different kinds of security challenges for public cloud platforms including data privacy, integrity, Multi Tenancy Security and Privacy and Access control. ENISA [7] also focuses on various security challenges that cloud computing is facing. Although various security issues still exist for cloud computing platforms, different non-profit and government funded organization have put a lot of their efforts to provide some security guidelines. In essence, NIST [5] and CSA [6] provide security guidelines to be followed for cloud computing environments.

Taking all these facts into account, the goal of this research paper is to design generic and secure architecture that can extend the security services in any cloud computing platform. Different kinds of computing platforms exist today, out of which some are proprietary and some are open source. Due to proprietary business of cloud platforms, the scope of this report is limited to open source (OpenStack) platform. However, our security extensions are modular and granular enough to be adopted by any cloud platform.
1.1. Background

Although researchers around the globe have different views while defining cloud computing, there isn’t any agreed single definition yet [12]. NIST [5] defines cloud computing as “A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models”.

The importance of cloud computing and its adoption can be best described in terms of its underlying characteristics, delivery and deployment models, how customers can use these services, and how to provide them securely. Cloud computing consists of three delivery models, four deployment models and five characteristics [4][5][12]. These models and characteristics lie on the top of each other, thereby forming a stack of a cloud. The three delivery models of cloud computing environment are: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS)[3][5][1]. Infrastructure-as-a-Service can be defined as virtual machines on demand, where users benefit from networking infrastructure facilities, computing services, and data storage. Amazon and Rackspace are leading vendors for IaaS platforms. PaaS is built on the top of IaaS, from where end-users can run their custom applications using their service providers’ resources. Examples of PaaS are App Fog, Google App etc. SaaS is build on the top of PaaS which provides delivery of business applications designed for a specific purpose. SaaS comes in two distinct modes named simple multi-tenancy and fine grained multi-tenancy. An example of SaaS is the SalesForce.com CRM application [3][13][1]. These delivery models reside at the second layer of cloud stack. In terms of deployment models, cloud computing platform includes public cloud, private cloud, community cloud, and hybrid cloud. Public clouds are predominantly owned by large scale organizations and services owned by this cloud are made available to the general public or a broad industry group. Private cloud is owned solely by one organization and is available for a particular group, while community cloud is shared and managed by the particular organization and supported by the specific community that has shared concern. Hybrid cloud is composed of two or more clouds (private, public, and community) [3]. These deployment models reside at the third layer of a cloud stack. The five characteristics of each cloud are: location-independent resource, pooling, on-demand self-service, instant
elasticity, transparent network access, and regular service [12][3][5]. They are located at the top of cloud stack. Figure 1 below represents the logical construction of cloud computing architecture.

![Cloud Architecture Diagram]

**Figure 1: Cloud Architecture**

### 1.2. Problem Statement

Cloud computing represents a paradigm shift of a traditional data center, IT management and a new summit of an IT capacity to drive a business ahead [29]. Despite its advantages, like on demand service, pay-as-you-go, resource allocation, etc, there exist critical security related vulnerabilities within the cloud computing platform. Some of them are listed below:

**Data Privacy and Reliability**

In a cloud environment the same data center may contain information belonging to different customers in an identical computer. In such cases information belonging to different customers is needed to be isolated, which further raises the issue of reliability. As system platforms of CSP (Cloud Service Providers) are shared among different customers, reliability may be an issue. For
example, malicious software or viruses may penetrate services and further affect user environment [14].

**Data Integrity**

Data integrity is important concern and it has been argued that should be strengthened in every cloud computing environment, as it is susceptible to both internal and external attacks. The lack of trust between CSP and cloud user may also raise the issue of data integrity. Unlike traditional database systems, all of its tenant’s information is stored on typical data center. C. Cachin, I. Keidar and A. Shraer [15] give illustration of risk and attack from both inside and outside of the cloud environment. For example, breach of data occurred in 2009 in Google Docs. More examples of attacks can be found in [15][16][17].

**Authentication and Authorization**

Web GUIs have become one of the most sophisticated technologies for delivering cloud services to both user's and administrators. However, they lack certain fundamental aspects of security for what is available on the front-end GUI. Some of its limitation in terms of authentication and authorization are:

1) Front-end GUIs are generally designed to ease the communication between cloud services and its backend components through API calls. These GUIs represent different functions of a basic management console. However, the whole architecture is designed and implemented using the corresponding structure as traditional web server with Internet access. Thus, it is vulnerable to potential attacks, unless certain countermeasures are applied.

2) Standard OpenStack front-end GUI uses username and password login method in order to access to the user dashboard. This approach has certain disadvantages as compared to other authentication frameworks, like OpenID[18], SAML[19], or Shibboleth[20].

3) Front-end GUI, being a separate OpenStack entity, requires maintaining separately the different user’s credentials. General concept of web services discusses this requirement; however, OpenStack still lacks federated authentication architecture. This means that users are first authenticated by the front-end component while suggest that backend components have no role in the authentication process. This is related to the concept of multiple PDPs (Policy Decision Point), contradictory to the fundamental concepts of security, which suggests that there must be single PDP, serving multiple PEPs (Policy Enforcement Point).
4) Authorization system varies in different cloud computing platforms. Some cloud platforms use identity-based authorization and access control, while some use other methods. But they all still lack federated access control and authorization.

**1.3. Purpose and Goals**

The purpose of this research was to study, analyze and extend security infrastructure of different components of the OpenStack cloud computing platform. The initial phase includes study of multiple cloud computing platforms. Due to proprietary business in cloud computing market, we opted to choose an open source platform. OpenStack, being an open source software and from the perspective of its architecture and future research, was selected as targeted platform for our security extensions.

The design is based on Service-Oriented Architecture and provides a seamless access to the services offered by cloud computing platform. Our design includes generic and secure architecture for front-end users, where all security services are provided in terms of web services defined by our security system. Moreover, the enterprises applications running behind the scene are tightly coupled. This includes design of central security server for large-scale distributed cloud environments. Central security server is responsible for generating certificates and for handling authentication (SSO) and authorization protocols. This includes active interaction with identity management server. The identity management server, apart from providing identity information to the central security server, can further be extended as a middleware for Keystone in order to provide federated identities.

**1.4 Research Methodology**

Design science methodology is selected in order to accomplish the goal of this Master Thesis, because our research is focused on designing and developing secure and generic architecture (artifacts) for cloud computing platforms, which address common security issues form single domain to multi-domain cloud platform. Our research methodology includes various steps, as described below:
• Identifying security issues in cloud computing platforms,
• Problem identification and design of secure architecture,
• Selection of appropriate open source cloud platform,
• Scrutinizing various component of cloud computing platform. In our case, OpeStack,
• Prototype implementation, system evaluation and analysis.

1.5 Thesis Outline
This Master Thesis is divided into nine chapters. Chapter 1 begins with introduction and background information about various cloud computing platforms, followed by problem statements, research goals and methodology. Chapter 2 discusses security of cloud computing platforms. This chapter focuses on security issues identified by non-profit and government organizations, like CSA, NIST and ENISA. Chapter 3 provides overview of security mechanisms for cloud computing platforms. This chapter discusses how common security mechanisms can be implemented in different cloud computing platforms, regardless of their models. Further, this chapter also discusses security mechanisms of different cloud computing platforms. Chapter 4 contains the core result of our research. This chapter describes about our security architecture and its security services. Chapter 5 discusses on open source cloud computing platform, namely OpenStack. This Chapter describes OpenStack platform and its components. Further, it also provides information on how we can extend security features of current OpenStack platform with the aid of our central security system architecture, described in Chapter 4. Chapter 6 discusses integration of our central security system with OpenStack platform. Chapter 7 describes small demo of our prototype implementation. Chapter 8 contains information on system evaluation of our Thesis work. Chapter 9 provides conclusive remarks of our whole research and suggests some future work for further research.
Chapter 2

Security Issue in Cloud Computing Platforms

2.1 Background

In order to identify security related issues in the OpenStack software, we first need to identify critical vulnerabilities in cloud computing platforms. Different non-profit and government organizations have put a lot of effort in order to identify security vulnerabilities and thereby providing the recommended solution.

In this chapter we will discuss security issues for cloud computing platforms provided by non-profit organizations that consist of industry representatives - Cloud Security Alliance (CSA) followed by two esteemed government organization, named National Institute of Standard and Technology (NIST) and European Network and Information Security Agency (ENISA).

2.2 Security Issues Identified By Cloud Security Alliance

Cloud Security Alliance is a non-profit organization, initiated by industry representatives in November 2008 and later supported by large number of IT companies, including Google, VMware, Microsoft, IBM, Ericsson, etc [25]. The main motive of this organization is to provide security assurance and education in the field of cloud computing [24].

CSA published its first draft “Security Guidance for Critical Area Focus In Cloud Computing” on April 2009 which provides information about security issue in cloud computing platforms. For our analysis, we use the current version (v3) of this draft. The guidance is divided into fourteen domains. The first domain named “Architectural Framework” gives brief information about cloud computing platform and its reference model from the security perspective. The rest of the domains are divided into top two categories named governance and operation. The governance category discusses “strategic and policy issues of cloud computing platforms” and operation category focuses “on more tactical security concern and their implementation within the architecture” [6]. Logical construction of security issues identified by CSA is described in Table 1:
Table 1: Security Issues identified by CSA

<table>
<thead>
<tr>
<th>Strategic and Policy Issues</th>
<th>Tactical Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Legal Issues: Contracts and Electronic Discovery</td>
<td>7. Data Center Operations</td>
</tr>
<tr>
<td>3. Compliance and Audit</td>
<td>8. Incident Response, Notification and Remediation</td>
</tr>
<tr>
<td>5. Portability and Interoperability</td>
<td>10. Encryption and Key Management</td>
</tr>
<tr>
<td></td>
<td>11. Identity and Access Management</td>
</tr>
<tr>
<td></td>
<td>12. Virtualization</td>
</tr>
<tr>
<td></td>
<td>13. Security as a Service</td>
</tr>
</tbody>
</table>

“Governance and Enterprise Risk Management” focuses on agility of an organization to govern and measure risks associated with cloud computing platforms. It also recommends that security department should be included during Service Level Agreement and contractual obligations.

“Legal Issues (Contracts and Electronic Discovery)” deals with legal issues associated with cloud computing platforms. The strategy and policy that is needed to be applied in a cloud in order to protect the information and computer systems, regulatory requirements, privacy requirements and international law to be followed by cloud providers.

“Compliance and Audit” focuses on compliance requirements for cloud computing platforms, such as regulatory, legislative etc, and its impact on internal security policy. “Information Management and Data Security” focuses on data manipulation, such as creation, usage, sharing, storage, deletion, and
“Portability and Interoperability” focuses on interoperability standards required between different cloud providers and also provides some recommendation to be followed by both deployment and delivery models of cloud computing platforms. “Traditional Security, Business Continuity and Disaster Recovery” focuses establishing traditional security functions, business continuity process, i.e. continuity of components of a cloud platform by assuring CIA (confidentiality, integrity, availability) and backup, disaster recovery process for cloud storage. “Data Center Operation” provides information on how we can evaluate the provider’s data center operation in order to select the best one for long term stability. “Incident Response Notification and Remediation” helps us to understand complexities, brought by cloud in current incident handling program. Further, it also addresses the necessary environment that is needed to be set up between both user and provider for proper incident handling and forensic. “Application Security” delivers the information on modern software development cycle that is needed to be utilized by cloud computing platform. Further, it also gives us information on security threats and vulnerabilities pertaining to cloud based delivery models (IaaS, PaaS and SaaS). “Encryption and Key Management” gives information on protecting access to data and resources. Further, it also recommends us to use OASIS Key Management and Interoperability Protocol for key management functions. “Identity and Access Management” focuses on importance of identity and access management in cloud environments. Further, also focus on federated identity and the problem faced by organization while extending its identity to cloud. “Virtualization” discusses security issues related to system hardware and virtualization technology. Some of the items covered in this domain are hypervisor vulnerability, risk associated with multi-tenancy, VM isolation and VM co-residence. Finally, “Security as a Service” focuses on open issues identified by CSA which include participation of trusted third parties for security assurance, incident management, compliance attestation, and identity and access management.

2.3 Security Issues Identified by NIST
National Institute of Standards and Technology is government funded organization in the US, continuously assisting cloud computing platform users by identifying security-related vulnerabilities in the platform. Security issues discussed by NIST are specifically focused to public cloud vendors, as it states that organizations have more control of each layer of security when private cloud deployment model is used. Unlike other government funded organizations
like, CSA and ENISA, NIST does not make any top level classification of security issues like, organizational, policy or legal. However, each issue discussed by NIST can be linked with the sub-issue identified by other organizations. Logical construction of security issues identified by CSA is described by Table 2.

**Table 2: Security Issues Identified by NIST**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Governance</td>
</tr>
<tr>
<td>2.</td>
<td>Compliance</td>
</tr>
<tr>
<td>3.</td>
<td>Trust</td>
</tr>
<tr>
<td>4.</td>
<td>Architectural</td>
</tr>
<tr>
<td>5.</td>
<td>Identity and Access Management</td>
</tr>
<tr>
<td>6.</td>
<td>Software Isolation</td>
</tr>
<tr>
<td>7.</td>
<td>Data Protection</td>
</tr>
<tr>
<td>8.</td>
<td>Availability</td>
</tr>
<tr>
<td>9.</td>
<td>Incident Response</td>
</tr>
</tbody>
</table>

Governance focuses on policies and procedures needed to be followed by organizational units. It also raises an issue of information security risks. Enterprise risk is due to lack of control of services offered by cloud and it recommended using auditing tools and risking management program. “Compliance” discusses the issues of data location, privacy and security controls, record management, and electronic discovery. The next section “Trust” discusses various topic and issues of internal threats caused by multi-tenancy, maintaining data ownership and intellectual property rights, risk management, gaining visibility and security control offered by CSP. The “Architecture” section discusses the issues pertaining to software systems utilized by cloud platform. Most of the issues discussed in this section are due to unique characteristics of cloud computing platforms which are completely different compared to traditional data centers. The issues covered in this section are hypervisor security, virtual network protection, virtual
machine images and client side protection. In “Identity and Access Management” the researchers from NIST focus on identity verification, authentication and access control mechanism and also recommend using SAML for authentication and XACML for access control. The next section “Software Isolation” warns about the threats associated with multi-tenancy such as the attack vector. Data Protection focuses on the need of data privacy and isolation, as data from different customers resides on common data center in cloud computing platforms. Availability section discusses about the threats that have a negative impact on organizational resources. Denial of service, equipment outages and natural disasters are some of the issue that is discussed. Finally, “Incident Response” section focus on reactive countermeasure for the attacks and threats in a cloud environment.

2.4 Security Issue Identified by ENISA:
The European Network and Information Security Agency is another government funded organization aiming to provide better security functionality in cloud computing platform. ENISA published its first document “Cloud Computing Benefit, Risk and Recommendation for Information Security” in November 2009. The document began with highlighting key benefits of security for cloud computing platforms. The rest of the document discusses security issues which are structured into three categories. All security issues discussed in each category are listed on the table below.

Table 3: Security Issues identified by ENISA

<table>
<thead>
<tr>
<th>Policy and organizational issue</th>
<th>Technical issue</th>
<th>legal Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lock-in</td>
<td>Resource exhaustion</td>
<td>Subpoena and e-discovery</td>
</tr>
<tr>
<td>Loss of governance</td>
<td>Isolation failure</td>
<td>Risk from change of jurisdiction</td>
</tr>
<tr>
<td>compliance challenges</td>
<td>Cloud provider malicious insider</td>
<td>Data Protection Risk</td>
</tr>
<tr>
<td>Loss of business reputation due to tenant activities</td>
<td>Management Interface compromise</td>
<td>Licensing risk</td>
</tr>
<tr>
<td>Cloud service termination or failure</td>
<td>Intercepting data on transit</td>
<td></td>
</tr>
<tr>
<td>Cloud provider acquisition</td>
<td>Data leakage on up/download, intra cloud</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Insecure or ineffective deletion of data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Distributed Denial of Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Economic Denial of Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Loss of encryption keys</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Undertaking malicious probes or scan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compromise service engine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conflict between customer hardening procedure and cloud environment</td>
<td></td>
</tr>
</tbody>
</table>

Policy and organizational issues cover six different issues present in a cloud computing platform. Lock-in discusses about data and service portability issue in terms of adoption of cloud service model. Afterwards, loss of governance and compliance challenges are remaining in this sub domains also discuss portability issues and its impacts on organization assets, risks and vulnerabilities.

Technical issues start with a list of threats present in a computing platform. Some of the threats discussed in this topic are availability due to resource exhaustion, VM monitor vulnerability, insider threats, denial of service, network related threats, and lack of sufficient effort from consumer to secure execution environment.

Legal issues begin with subpoena and e-discovery issues, which provide information on how to respond subpoena and e-discovery issues. The rest of the legal issues discussed in this section are focused on data manipulation, data location compliance, data protection compliance and risk of losing intellectual property, when data is stored in a cloud.
Chapter 3

Overview of Security Mechanisms for Cloud Computing Platform

3.1 Background

This chapter provides a brief overview of security mechanisms in cloud computing platforms followed by security mechanisms that are available and can be deployed in various cloud computing platforms. Finally, this chapter also discusses security mechanisms in cloud computing platforms based on their delivery and deployment models followed by available security products and modules.

3.2 Security Overview

Cloud computing is a paradigm shift of technology that have emerged and has been adopted by many IT organizations in the recent year. This shift in technology has changed the overall architecture and system requirements, compared to traditional server-based systems. Cloud-based system architecture provides Internet-based services, computing and storage in all fields including health care, finance, government etc [29] with the reduced price. Therefore, it is more likely to be adopted by most of the IT organizations. While there is a serious concern for an organization to move towards the cloud based service, security risk associated within the platform are one of the urgent concern for an organization to make this move. Different cloud based deployment models have brought the wide range of security risks and concerns that have to be evaluated and mitigated [30]. Traditional defense in depth security model which include physical security, perimeter security, firewall, antivirus software, etc are not directly applicable to cloud-based systems. This means that various organizations must adopt the best security practices and standards that are somehow incompatible to traditional defense in depth security models. However, the same principle of multi-layered security is still applicable.

In cloud-based platforms, no matter what we choose as our deployment model, we will be working with abstracted and virtualized environment. This means that software or platform will run on the top of shared physical infrastructure, which is managed either internally (in case of private cloud) or by CSP (in case of public cloud). This means the security architecture for application services will need to shift from platform to the application layer. For example, if application provided by the cloud vendor is Software as a Service, then the end-user have no
control over the software development platform or infrastructure used and thus all the security countermeasures are needed to be placed at the application level.

3.2 Security Mechanisms for the Cloud Service Model
Cloud-based system addresses three service models named IaaS, PaaS, SaaS. These service models lie on the top of each other, thereby forming the stack of a cloud. The IaaS service model can be deployed using one of the deployment models, as discussed on Chapter 1. Hence security implications need to take into account considering both service and deployment models.

3.2.1 Security Mechanism for IaaS
Infrastructure-as-a-Service, sometimes also referred as utility computing, can be viewed as virtual machine on demand, where this virtual machine can be accessed remotely and made available on elastic basis. This means that all the necessary infrastructure, hardware, memory, networks and storage are provided by IaaS service model. Most of the security concerns for IaaS delivery model are due to sharing and pooling of resources, virtualized data center, and virtualization of hardware, resources and networks. No matter what we opt to choose as our deployment model for IaaS, security requirements for IaaS service model must be implemented at the level of host, virtual machine, network, storage, compute and memory.

For public/hybrid cloud model, all the cloud services are provided to cloud service clients via Internet. The cloud service client in this case may be client computer or any on-premises system that is connected to cloud-based IaaS system. Depending on the services offered by cloud-based IaaS system, we can control security state of the client system connected to cloud services. This can be done by enforcing the baseline security level of all clients to assure that the client has sufficient security tools, like anti-virus, anti-malware and up-to-date security patches. However, if these cloud services are available to an unaffiliated user, there is nothing that IaaS vendor can do to enforce security policy of this non-affiliated client. Therefore, system must be designed to support the level of network encryption or even in the worst scenario secure session must be established for the logging process.

Another issue associated within this delivery model is network availability. Attacks like DNS misdirection, Prefix hijacking, or distributed denial of service can seriously deteriorate the network availability of the system. Therefore, constant network monitoring and auditing tool must be implemented to mitigate the attacks on network availability. Moreover, in a
private/hybrid cloud network resources are consumed from a common resource pool. Consequently, logical isolation of internal system is equally important for cloud-based IaaS system. This means that in order to secure the network system of our cloud-based IaaS delivery model virtual firewall, VLAN, virtual layer 2,3 switches, and IPSec isolation are needed to be considered and implemented.

Another issue associated with IaaS delivery model is due to its architectural representation of its cloud-based storage system. Cloud based storage system is designed for creating the pool of resources, abstracting the details like storage location, storage type, persistence of storage, etc. from its consumers [31]. This means that data from multiple tenants reside on same disk or array and any breach in the system could lead to the exposure of private and sensitive data to a malicious user or unintended tenants. An appropriate access control (XACML) and authentication (SAML, OpenID) mechanisms in terms of identity of the user can mitigate this issue. The implementation details of this access control and authentication mechanisms depend on the deployment models of cloud-based platforms. Public cloud, offering IaaS delivery model, may use web services through web portals to provide access control and authentication mechanisms. Hybrid cloud may use public cloud storage gateway appliance on premises, where the cloud storage API is translated into conventional data retrieval protocol, like ISCSI, NFS (Network File System), SMB (Server Messaging Block), etc. [31].

### 3.2.2 Security Mechanism for PaaS

Platform-as-a-Service model is build on the top of IaaS, which provides complete development environment where application developers can create and deploy their applications. In contrast to traditional software development tools, like Visual Studio, PaaS offers a shared development environment. This means that there must be a mechanism within the system to ensure that customers are kept separate from each other. An appropriate authentication, access control and authorization mechanisms will ensure isolation of customers. A strong and implicit authentication mechanism ensures that user is correctly identified. Most of the Paas providers still rely on the same traditional user-name and password-based authentication and then apply access control and authorization mechanisms based on verification of the credentials provided. An alternative to this, two factor authentication mechanisms, like smart cards and biometrics, can be implemented. Moreover, identities-based authentication in terms of web services or SAML-
based identity provider can be taken into account, where authentication authorization and access control in a PaaS system can be externalized.

### 3.3.3 Security Mechanism for SaaS

Software-as-a-Service is built on the top of PaaS, and all of the security mechanisms are implemented at an application level, regardless of its deployment model. Network security is typically not considered in the SaaS delivery model. However, it can be implemented with regards to some application specific control of SaaS solution. In a Public cloud scenario, high degree of trust in the cloud vendor is required, as the infrastructure and platform are under the supervision of cloud vendors [33]. Therefore, security responsibility of both cloud vendor and customer are defined in the Service Level Agreement (SLA). Another factor to be considered for a SaaS-based solution is its APP's store. Cloud vendor may offer their application only from their app store, and there is a possibility that any malicious user can post malware in the app store. Google Android had a similar problem in the past [32]. Moreover, one must also assume that SaaS-based software solution will be scanned by the hacker in order to identify the vulnerabilities before deploying it. Due to the absence of App's store in a private cloud, threats associated with malicious malware from outside of the company are no more the area of concern. However, App's developed with poor piece of code can be as detrimental as malware. This means that regardless of deployment model on a cloud-based platform, security guidelines for developing software based solutions, like SDL (Security Development Lifecycle), should be followed before developing SaaS-based solution in a cloud.
Chapter 4

Design of the Cloud Central Security System

4.1 Background
This chapter discusses the architecture of our central cloud security system which is designed for delivering “Security-as-a-Service” model to the cloud stack. Our central security system is based on the facts that by shifting all the security related services to application level, a generic and secure framework for cloud-based platform can be deployed. This means that all the security related services, like identity service, SSO and identity and access control, authentication and authorization mechanisms, are provided by our cloud security infrastructure.

4.2 Security System Architecture
Security system architecture is motivated by the discussion presented on Chapter 2 and Chapter 3, where all the security related services for cloud-based platform are shifted form a platform to an application level and are provided as web services by our security system architecture. One of the advantages of shifting all the security-related service to an application level is based on its design modularity and generosity. This means that our architecture is applicable to any cloud-based platform, regardless of its delivery and deployment models.

The components of our security system are based on “Service Oriented Architecture” and are responsible for managing and distributing certificates, identity management (CRUD), identity federation, creating and managing XACML-based policies, and providing strong authentication mechanisms. All the components within the system are interoperable and act as a security service providers in order to assure a secure cloud-based system. Figure 2 shows logical components of our central security system.
PKI server, also known as Local Certification Authority (LCA) in our system is responsible for issuing and distributing X509 certificates to all components in a domain. This server can either be configured as single certification authority, by generating self-signed certificates or may be linked to PKI in order to exchange certificates and establish trust relationship between various domains. In this case higher level trusted certification authority server issues certificates to the issuing CA. XACML server is also known as Policy Decision Point and is responsible for creating and validating SAML Tickets for Single Sign-On protocol. This server is also responsible for management of group, roles, XACML policy and policy sets. IDMS server is responsible for creating, reading, updating and deleting identities in a collaborative environment. Strong Authentication (SA) server performs mutual authentication with clients using various extended authentication protocols, like FIPS 196. This server also interacts with XACML policy server to generate SAML ticket for authenticated clients [28].

4.3 Cloud Security Infrastructure
Figure 3 shows logical structure of our cloud security infrastructure. The infrastructure is based on the assumption that the application service providers will focus only on their business logic rather than implementing built-in authentication and authorization services. This means that all security services are isolated and shifted towards the central security server, controlled by central security administrator. SAML and PDP system entities are responsible for delivering identity
services to application service providers in both secure and interoperable manner. As shown in Figure 3 end-user (may be Enterprise Administrator) interacts with Cloud access point through Internet. The Cloud access point typically behaves as cloud entry point for our cloud security infrastructure. To have a secure communication, i.e. exchange of messages, cryptographic services like message encryption and digital signature, are required by the end-entities. This means that in order to protect the resources, the system must be facilitated with public and private key pair. Since many enterprises business, resources and applications run behind the access point in a cloud environment, access point must be preserved by some secure authentication mechanism. Implementing PKI can fulfill our objectives.

![Central and Portal Security Servers](image)

**Figure 3: Cloud Security Infrastructure**

### 4.4 Security Services offered by our Central Security System

The overall approach to a design of a central security system is to provide secure means of communication for all end-users and application services running in a cloud environment. In
In essence, our design for providing security services is based on Service-Oriented Architecture and is defined in terms of web services. Security services offered by our central security system are authentication, authorization, identity management, access control, and SSO. Further, it is also assumed that Strong Authentication server and local certificate authority have already been designed and implemented.

### 4.4.1 Single Sign-On Protocol

A single enterprise business, running in a cloud can provide more than one application to its end-users. All of the application services should authenticate clients before service transaction are executed. This means that as number of application grows, so do the number of security credentials (logins URLs, username and password). Unfortunately, having many security credentials for authentication purposes is mostly unlikely from security and system coordination and management perspective. As cloud applications are adoptable and growing in large scale, it becomes a major requirement to provide SSO service to its end-users.

The SSO service is offered by the central security server from our cloud security infrastructure. SAML server provides SSO service to application providers by providing SAML ticket which provides assurance of client identity verification for authentication purpose. Once client is authenticated, resources authorized to authenticate a client are available without the need to re-authenticate for each domain. In order to achieve the SSO service, the other components of our central security server must coordinate and interact with each other. This means that all application services and other three components of our central security system must be registered in our IDMS system in order to provide SSO service by SAML server. As shown in Figure 3, when the user wants to access resources from application service providers for the first time, the user is redirected to the central authentication server. Central authentication server, which acts as proxy server for all service providers in our central security system, is responsible for managing authentication procedure and identity verification. This means identity of the user is first verified by IDMS server and upon successful identity verification, user X.509 certificate is verified by local certification authority server. The result of this authentication procedure is then passed to the SAML server, which issues a SAML ticket based on the credential (X.509 and identity), passed to the SAML server. The SAML ticket is then passed to end-user through Authentication Server. This SAML ticket is later embedded in a request directed towards the application service providers and has a validity period. The period up-to which the SAML ticket remains valid
depends on the organizational policy. A valid local session is created in order to successfully authenticate user, so that user can request services from other application service providers with the same ticket until the ticket expires. The whole mechanism is based on the assumption that there is a trust relationship between the SAML service provider and application service providers existing in different security domains.

### 4.3.2 Identity and Access Control

As mentioned earlier, cloud-based platforms are capable of hosting different application services of application service providers using the same physical resources. No matter what amount of resources application service providers are consuming for their services, each application service must be logically separated and there must be the mechanism of user provisioning, deprovisioning and overall life-cycle management of user and access in an automated fashion. One of the approaches to handle access control mechanism is to allow each application service provider to implement independently access control mechanism by means of self-governing security policies and policy enforcement points. However, the overall approach to implement independent access control and PEP seems to be fairly complex and expensive and it is not suitable for multi-domain cloud-based platform. This means that there must be an efficient way to handle identity and access control mechanisms in our cloud-based system. Moreover, with the emergence of IT in cloud-based platform, IAM in a cloud computing environment is not confined to a single domain where identity and access control mechanism of different enterprises and IT organization are needed to be considered. This means that the overall system must be architected in such a way that it must be able to provide flawless identity management, access control and identity federation. Using IAM as a web services in a Service Oriented Architecture for our cloud security infrastructure fulfills the overall requirement of identity federation and identity and access control mechanisms.

The overall approach to identity federation, based on SSO is provided by the fact that each service provider must be registered in our IDMS server. After that, based on valid credentials, each service request is processed, validated and authorized. Access control mechanism, often referred as authorization, is based on role defined by the entity of our central security system. A single Policy Decision Point (XACML) server is responsible for entire authorization process. Having a single PDP as component in our central security server optimizes the authorization process in more flexible and secure way, as it can be managed, configured administered and
protected separately from application services. The PDP server supports management of groups, roles, XACML policies and policy sets defined by security administrator, based on which authorization and access control mechanism is processed. Application services are protected by PEP and can be implemented either as a separate service for each application service or integrated with application server. Figure 3 shows logical construction of the authorization process. End-user requests access to the resources, the PEP intercepts the request and creates SAML authorization request. The authorization request which contain resource, action and role of the user is then sent to the PDP Server. Based on policy and policy set, defined by security administrator, PDP server evaluates the request and sends authorization response back to the PEP. Based on the evaluation made by PDP, PEP service grants or denies the access to the requested resources.
Chapter 5

Security Extension on OpenStack

5.1 Background
This chapter describes background information on security of OpenStack project in order to understand the goals of our Thesis. The first section describes the overview of the OpenStack project and its components. The second section describes our motives for choosing OpenStack for this Master Thesis research, followed by our deployment strategy for OpenStack and needed security extension features of current OpenStack deployment.

5.2 Overview of the OpenStack Project
OpenStack is open source software, which comprises the collection of open source technologies in order to build both private and public clouds. OpenStack software consists of four novel technologies in order to provide massively scalable cloud computing operating system. Rackspace and NASA are the initiators of the OpenStack project and it is continuously supported by a different organization, like Citrix, Dell, IBM, Cisco etc.

5.3 OpenStack Components
Current version of OpenStack software consists of five individual components, each component is an open source project and all are developed and nurtured under the jurisdiction of OpenStack community. The five components of OpenStack project are:

1) OpenStack Compute (Nova): OpenStack Compute is a heart of whole OpenStack Project. It provides a tool to orchestrate the cloud [8]. It is used to provision and manage large networks of virtual machine (Iaas) for creating massively scalable cloud computing platform and is similar in scope to Amazon Ec2 and Rackspace cloud servers. OpenStack Compute does not include virtualization software. However, it defines a driver to interact with underlying virtualization technology like Xen, KVM etc. The logical structure of OpenStack Compute (NOVA) is shown in the figure 4.
2) **OpenStack Image Service (Glance):** OpenStack Image Service (Glance) is a service catalog for discovering, registering and retrieving virtual machine images [21]. Glance consists of three novel components named, Glance API's, Glance Registry, and Glance Image Store. Glance Image Service provides a REST API service for end-user to query the information of virtual machine metadata, as well as it retrieves actual image. The VM's available by Glance can be stored in a variety of locations including OpenStack Object Storage System (Swift). Glance also supports a variety of disk and container formats including OVF, BARE, ISO, etc [22]. The structure of OpenStack Image Service is shown in Figure 5:
3) OpenStack Object Storage (Swift): OpenStack Object Storage is used for creating scalable redundant object storage. Logical representation of Swift can be structured into two logical components: Presentation and resources [22]. Presentation logic is responsible for interaction with end-users via Swift proxy and also optionally used for authentication and authorization. Resources are typically used to manage information sources through three processes in order to fulfill the incoming request from Swift proxy. The overall structure of Swift is shown in Figure 6:
4) **OpenStack Identity Service (Keystone):** Keystone is OpenStack project for providing identity, token and policy services for other OpenStack projects. Keystone implements OpenStack Identity API in order to provide a valid token for its tenants and users. The introduction of this novel projects to OpenStack family not only can provide Identity-as-a-Service to cloud delivery model, but can also be integrated with our cloud security infrastructure, where all the security services are provided by our central security system. The structure of Keystone is shown in Figure 7:
5) **Horizon**: Horizon is the canonical implementation of OpenStack Dashboard for providing web based interface to other components of the OpenStack [23]. The key features of Horizon are extensibility, manageability, usability and support for other OpenStack projects. The structure of OpenStack using Dashboard is shown in Figure 8:
5.4 Motivation for using OpenStack

There are several open source cloud computing platforms that provide Infrastructure-as-a-Service delivery model to both private and public clouds. OpenStack and Eucalyptus are example among many other available platforms. The main motives of choosing OpenStack for our research are due to its distinctive features. Some of its features are:

1) Modular middleware architecture: This means that each module of the OpenStack project can be treated as a separate entity. This architectural modularity in middleware possesses significant advantages for extending current OpenStack modules where, each OpenStack project can be extended by treating it as a separate entity without affecting the overall system operations. This also means that we can design and integrate our design in the current OpenStack project. For example, extension of authentication and identity modules.

2) Authentication Module: The aim of this research is to implement strong authentication framework. This implies that the authentication module within the OpenStack project must be a
separate module. Current OpenStack project fulfills this requirement by providing architectural modularity of its authentication middleware. Hence, it was very appropriate for our research work.

3) Identity Module: Currently OpenStack project uses Keystone middleware as identity module. Again architectural modularity of middleware of the OpenStack projects provides a significant possibility to extend Keystone middleware to provide federated identity and access control mechanism within the complete OpenStack project.

4) Separation of GUI from back-end cloud servers: The evolution of Web services in cloud computing infrastructure totally differ completely from traditional Web services. In a cloud computing environment, all physical servers are located in a back-end what means that Policy Decision Point must be located in a back-end too, which is not always true in cloud computing platforms. This means that we require certain level of isolation of front-end GUI from back-end modules. OpenStack is complying with this requirement what makes our work easier.

5.5 OpenStack Deployment Strategy
Our OpenStack deployment uses one computer installed with Ubuntu 12.04 LTS. In our deployment one Ubuntu machine acts as Control and other as Compute node. The IP address provided for the server is used as a router. Further, we use single NIC for our OpenStack deployment strategy. KVM is used as Hypervisor, Flat DHCP is used as networking options for our guest VM and, finally, fixed range IP address is given to our guest VM which is connected to host via br100
5.6 Security Extension of OpenStack Platform

OpenStack is purely Iaas delivery model which can be deployed in any of the the available deployment models. If OpenStack is deployed as a private cloud delivery model, than as a cloud vendor we have security control from layer 1 through layer 7. But in case of public and hybrid clouds, all security services are available through Web Services residing at an application level as discussed in Chapter 3. Regardless of the deployment model and in order to provide granular and modular security extended features, the extended security features for the OpenStack are provided by our central security system.

5.7 Current Authentication Mechanism in OpenStack Platforms

Current deployment of OpenStack uses Keystone as a central security hub [26] for authentication. The authentication mechanism is based on username and password, which are sent in clear text, and upon successful authentication, Keystone server generate valid token, which is used by end-user to take advantages of other services. Moreover, all the communications are performed using HTTP protocol. Therefore, it raises a certain security issues
as defined in Chapter 3, and are needed to be resolved. Figure 10 below describes current OpenStack authentication mechanism.

![Authentication Mechanism with Keystone](image)

5.8 Problems with OpenStack Authentication

From discussion in section 4.2 and [5], [6] and [7] we were able to identify problem associated with the current OpenStack authentication mechanism. Passwords are used for authentication and they are sent in clear text. Communication protocol used for authentication is HTTP, which is considered to be less secure than HTTPS. Authentication token generated is not a certificate and is only 32 bit long, what is considerably smaller, compared to a signed token signed by Certificate Authority. Moreover these tokens are short lived, i.e. they are valid for 24 hour only, which means that users have to authenticate themselves again after the validity of token expires.

With the implementation of the PKI in Keystone, we can generate certificate that are valid for longer time. Upon successful authentication and token validation, each service (Nova, Glance, and Swift) must to invoke Keystone in order to verify the authenticity of the token, which raises the issue of scalability. The randomness of the 32 bit token generated is still a question. As a Single point of failure, Keystone is a bottleneck for both authentication and authorization.
5.8.1 Extending OpenStack weak Authentication Using PKI

PKI can solely be implemented for Keystone. However, revocation and scalability are important issues. This means that a separate Certificate Authority, i.e. “Central Security Server” is needed that can provide a valid certificate to Keystone for handling certificate requests. Moreover, authentication with Keystone in the OpenStack is a two-step process. The first step involves the use of user-name and password and, upon successful authentication, Keystone generate a valid token. The second step involves the use of this valid token to provide SSO and delegated authentication [27]. Implementing PKI can improve the security of first step, what can further improve the security and scalability of the second step by providing signed authentication and authorization tokens.

5.8.2 Initial Authentication

Once the user is registered in the Keystone, OTP (one time password) generated for the user can be used to establish a key-pair. Public key is first signed by CA and stored in the Keystone server and Private Key is stored by the end-user system. The overall approach of the initial authentication process is shown in Figure 11:

![Figure 11: Initial Authentication in Keystone using PKI](image-url)
5.8.3 Delegation and Scaling

Keystone requires active participation of users for on-line validation of tokens and tenants, but with the implementation of PKI we can improve security and scalability of tokens that contain authentication and authorization information, i.e. tenant, set of roles that user have for this tenant, and username. This means that by cryptographically signing and encrypting information content of the token with Keystone's private key, we can assure that only the services which have Keystone public key can decrypt the messages. Thus, we are reducing the need of networks call for token validation and the need to go back to Keystone to fetch roles. The overall approach involves modification of Auth-token middleware to be a CMS (Cryptographic Message Syntax) based token. The delegation of authority with this approach can only be performed within the OpenStack deployment. Since signing of messages requires user's private key, which is maintained outside of Keystone system and current browsers do not support the means to sign the message on behalf of a website, delegation of authorization is required by Dashboard on behalf of logged-in user. This means that the user must first login to the Dashboard by using his/her certificate and then use Keystone token to delegate the authority from the Dashboard to other services of the OpenStack cluster.

![Authentication and Delegation Protocol](image)

Figure 12: Authentication and Delegation Protocol
Chapter 6

Integration of the OpenStack with the Central Security System

6.1 Background
This Chapter describes the information needed to integrate current OpenStack project with our Central Security System. The first section discusses how we can integrate out identity management system with Keystone. The following section describes how we can provide separate access control and authorization, i.e. SAML and XACML, to provide SSO to the Openstack environment. Finally, integration of the LCA is also discussed.

6.2 Integration with IDMS
Current OpenStack project uses Keystone identity back-end module for managing and distributing identity information. This means that Keystone identity modules are responsible for performing CRUD (Create, Read, Update, and Delete) operations for user management. However, as discussed in Chapters 3 and 4 separating identity information from the front end-users and thereby providing identity services through our Central Security System is the goal of our research. This means that CRUD operations for user management are provided by our central security system to the front end-users in terms of Web Services and hiding all the Keystone identity back-end modules from front-end users. This ultimately demands the needs for separate identity providers where IDMS from our Central Security System plays the role of the identity provider. This means that, whenever a user wants to use our cloud environment, the user must first register themselves in our IDMS. Once the user is registered, an API call to Keystone identity back-end module is made in order to register the user into Keystone database. Moreover, the same user information including user's 32 bit unique and random ID, as the response from Keystone, is also stored in our IDMS database. Our IDMS is based on SCIM stranded, which provides registration of users. Once the user is registered, then based on the credentials as ID, email or attributes is transfer to Central Security Server (Authorization Server), which will assign the role to the corresponding user and provide Authorization SAML ticket to the user for access to the OpenStack resources.
6.3 Authentication based on Certificates
When Local Certification Authority server is configured as Keystone authentication back-end module, the overall approach to the certificate based authentication is as simple as discussed on Chapter 5. However, LCA server and IDMS server must interact with each other in order to assure valid user authentication based on certificates. Local Certification Authority server is at the bottom of the PKI hierarchy. Client registered in our IDMS server are responsible for generating key-pair, where public key of the client is used to create the certificate request. The certificate request is made to LCA which creates the certificate for the client if the identity of the client is verified by our IDMS server.

6.4 Authentication/SSO based on SAML Token
Keystone and other projects of OpenStack provide an interfaces and API to integrate SAML protocol for SSO into OpenStack environment. This means that once necessary libraries and plug-ins are built for Keystone identity back-end module, authentication/SSO based on SAML token is pretty much similar to the discussion in Chapter 4 on section 4.3.1.
In order to provide authentication/SSO based on a SAML token to the OpenStack environment, all application services and components of our Central Security System are needed to be
registered in our IDMS. This means that for each service request made by end-user, identity verification of end-user is made by IDMS server. After successful identity verification user X509 certificate is then verified by our LCA server. The result is then passed to the SAML server, which issue SAML ticket based on the authentication and identity information passed to the SAML server. SAML ticket consists of user meta-data, i.e. username and password, Tenant information, role and a valid session period is redirected to the Keystone identity back-end through a secure communication channel established between the service request and Keystone’s back-end modules. Keystone identity back-end module verifies SAML ticket and in response provides a 32 bit random unscoped token back to the client, which is used by client to access end-point services.

6.5 Authorization based on the XACML Policy
Authorization based on an XACML policy is based on the assumption that all components of our Central Security System are fully organized and function in a collaborative environment. This also means that necessary configuration and interfaces for Keystone identity back-end module are synchronized with the components of Central Security System. For the authorization purpose, we have chosen a single Policy Decision Point that is responsible for handling the entire authorization processes. PDP server performs management of groups, roles, XACML policies and policy sets defined by the security administrator, based on which authorization and access control mechanism is processed. This means that whenever the end-user requests some services, PEP intercepts the request and creates XACML authorization request. The request contains resource (Tenant information) action, and the role of the user. Based on the policy and policy set, defined by security administrator, PDP server evaluates the request. Evaluation response is sent back to the PEP. PEP Server now grants or denies the request, based on the evaluation response made by the PDP.
Chapter 7

Prototype Implementation

7.1 Background

This chapter describes prototype implementation our Thesis work using OpenStack Platform. The purpose of this prototype implementation is to demonstrate and test the functionality of identity and authentication services, based on one-time password and PKI in Keystone. Our prototype implementation is based on the information provided in [26], [34] and [35]. As discussed in Chapter 5, in order to implement PKI in the Keystone Web server must first be SSL configured. This means that our Local Certification Authority is responsible for providing the certificates for both Web server and the Keystone.

7.2 IDMS Server Implementation

IDMS server is a separate entity for performing CRUD operations with end-user data and is one of the active components of our Central Security System. IDMS server is also responsible for registering other three components of our Central Security System. Our IDMS server is based on SCIM standard and follows standard protocol as discussed in Chapter 4. IDMS server, apart from providing identity verification, interacts actively with Keystone identity back-end module, so that any user registered in our IDMS server is also registered in the Keystone database. For this, an API call to Keystone identity back-end module is made, whenever user wants to register himself in our IDMS server. However, prior to the API call to the Keystone back-end module, user role must be defined by our PDP server. This means that user attributes like username, email, etc. are passed to the PDP server, based on which the PDP defines the role of the user for authorization purposes. Once IDMS and PDP servers are synchronized, they are ready to make an API call to the Keystone identity back-end module. The response of the Keystone API call is then stored in our IDMS database, which includes username, password, role, tenant information and, 32 bit unique random ID along with other necessary and optional parameters needed in the user registration process. The runtime implementation of the IDMS server for user registration is shown on Figure 14:
7.3 Combination of Keystone and PKI

The implementation of PKI is based on public and private key-pair using X509 technology. Here Keystone is responsible for storing Private and Public Keys and certificates. Once the system is setup, as described in section 5.7 and 6.1, any end-user can get certificate from the Keystone by calling its REST web services. However, certificates are not exposed outside of the Keystone service. This means that, when the services are invoked in PKI mode, each endpoint service gets public key from the Keystone which is saved by the services for later use. Now, when the user is authenticated using one-time password, Keystone builds the token in a PKI mode. This means that Keystone creates token JSON object which contains token's metadata, which is further encrypted using Keystone’s private key and then signs encrypted token using MD5. Any endpoint services which have Keystone public key can now decrypt and verify the token. Moreover, since the encrypted token includes metadata information, there is no need for the endpoint service to invoke Keystone call to get token's metadata. Hence, this approach is providing both scalable and delegated authentication protocol. Runtime authentication of user with Keystone using OTP is shown in Figure 15.
Figure 15: Authentication using One-Time Password in HTTPS

Figure 16: An Encrypted and Signed Token Provided by Keystone in PKI mode
7.4 SAML Implementation

Keystone could be modified to use SAML, but with the cost of replacement of many components, not just a plug-in upgrade for authentication. Having said that, Keystone and other OpenStack components have reasonably well-defined APIs and interfaces, which can be used to integrate it with the PDP server for SAML implementation. This means that a plug-in which communicates with our PDP server through Keystone back-end, has to be built on the top of current Keystone Projects. Due to limited time available for this Master thesis and insufficient information and knowledge on Keystone back-end modules, we left SAML implementation for further research. However, the PDP server is still capable of assigning the role to the user registered in our IDMS server. Runtime implementation of PDP server for assigning roles is shown in Figure 17.

Figure 17: User Role Registration
Chapter 8

System Evaluation

8.1 Introduction

System evaluation is based on the design and implementation of our central security system architecture. The overall system is evaluated in terms of scalability, usability and security, which is based on the assumption that all the components of our central security system are fully functional. Moreover, the details of reference implementation of the SAML and IDMS servers for our cloud computing platforms are available in [36].

8.2 Evaluation of Usability and Scalability

By shifting all of the security related services to the application level, our central security system not only offers the generosity in its architecture, but also enhances the usability of the security services offered by the system. All of the security related services are offered by means of web services from our central security system and are based on standardized protocols. This means that services offered by our central security system can be integrated with the components of the OpenStack cloud computing platforms. OpenStack uses Keystone as a main point for identity and access control, hiding all the Keystone services from front end-users. By integrating services, like SSO (Single-Sign-on) and IAM (Identity and Access Management), offered by our central security system into the OpenStack platform we not only assure system interoperability, but also deployability and usability. Moreover, introduction of the LCA (Local Certification Authority) server and integration of the PKI with Keystone, as discussed in Chapter 5, increase scalability of our OpenStack computing platform.

8.3 Evaluation of Security

Evaluation of security is based on security services provided by our Central Security System. Table 4 shows security threats that are needed to be considered by our Central Security System.

1) Authentication/SSO Evaluation: SAML ticket issued by the SAML server is transmitted over the public network, i.e. Internet to the end-user and from the end user to the application services for SSO purpose. This means that there must be mechanism to protect ticket exchange against the threats like replay attack, message modification, man-in-the-middle attack, impersonation, and repudiation. Moreover, application service
provider relies on the decision made by PDP server. This means that there must be pre-established trust relationship. The trust relation can be established based on PKI model, i.e. by digitally signing the response message generated by SAML server. Application service provider than verifies the signature of the response message by using root certificate for that domain. In this way we can provide authenticity and integrity of SAML tickets thereby thwarting threats, like message modification and repudiation. Moreover, the exchange of SAML ticket takes place over a secure communication channels using SSL, thereby, preventing against impersonation and replay attack.

2) **Authorization Evaluation:** As mention earlier in Chapter 4, all application services communicate with authorization service (PDP) through public network using request response protocol. PDP services are invoked form our Central Security System which has a trust relationship with other components of the Central Security System. However, it still can be subjected to threats like message modification, impersonation, man-in-the-middle attacks, and repudiation. This means that the communication channel and policy administration procedure should be protected. Implementation of a secure communication channel is achieved through SSL for our proposed architecture. Moreover, PEP and PDP mutually authenticate before processing any transaction and all XACML request-response messages are transmitted over a secure channel. In order to thwart the replay attack a randomly generated session ID unique per each session is used. Finally, trust relationship between each component is based on PKI model which ensures authenticity of XACML request-response from and through PDP and PEP Servers in order to provide valid authenticity and integrity of XACML request-response messages. This protects the entire process from message modification and impersonation attacks.

**Table 4: System Security Threats Evaluation**

<table>
<thead>
<tr>
<th>Security Services</th>
<th>Threats</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Impersonation</td>
</tr>
<tr>
<td>Authentication/SSO</td>
<td>x</td>
</tr>
<tr>
<td>Authorization/XACML</td>
<td>x</td>
</tr>
</tbody>
</table>
Chapter 9

Conclusions and Future Work

This chapter provides the conclusive remarks of our investigation during this master thesis projects and also recommends for future works.

7.1 Conclusions

Design and implementation of a generic and secure architecture for cloud computing platform is still an open issue in the field of security for IT organizations. Due to the varying nature of computing platform, in terms of delivery and deployment models, cloud still needs generic and secure architecture in term of its adoption. This Master Thesis is focused on design and implementation of a generic and secure architecture for cloud computing platforms. The whole architecture is based on the concept of Service-Oriented Architecture that can be deployed on any computing platform, regardless of its deployment and delivery model. OpenStack being an open source platform with modularity in its architecture makes our work easier for testing and deploying central security system architecture.

Based on our prototype implementation during our research we were able to conclude that by shifting all of the security services to application level, a secure computing platform can provide services like Single Sign-On, Identity and Access Management and certificate-based authentication. Moreover, all security related services, offered by our central security system, are delivered in terms of web services, thereby, they possess significant advantage in terms of their usability, deployability, interoperability, and scalability for any computing platform. Nevertheless, due to modularity in architecture of our central security system, any desired component can be added or modified, so that any further research can be processed without affecting the overall system performance.

7.2 Future Work

This research was focused on designing generic and secure architecture for cloud computing platforms in order to preserve identity information and deliver seamless access control and authorization mechanisms to the end-user regardless of cloud service and delivery model. Still there is a need to do more comprehensive observations and activities within this area. Some of them are listed below:
1) More work could be done in the area of identity federation, users privacy, and anonymity.
2) System architecture proposed has only been tested and deployed for IaaS platform. More observation and research can be conducted for PaaS and SaaS platforms in order to verify the applicability of our architecture.
3) The overall performance of the system can also be evaluated in a scalable environment in order to identify throughput and latency of the system. Moreover the uptime and downtime of the system can be also be evaluated, and finally
4) The unfinished work for SAML implementation can also be further processed.
Bibliography


55


[34] https://github.com/OpenStack/Keystone


[36] Cloud Single Sign-on and Authorization Services, Davit Hakobyan, Degree project in Second cycle, Stockholm, Sweden 2012, KTH (Royal Institute of Technology)